Haaga-Helia
University of Applied Sciences

Development of a wireless in-flight entertainment system for the airline industry

Alisher Aliev

Bachelor's Thesis
Degree Programme in Business Information Technology
2020

Haaga-Helia
University of Applied Sciences

Abstract

28 April 2020

Author
Alisher Aliev

Degree programme
Business Information Technology

Thesis title
Development of a wireless in-flight entertainment system for the airline industry**.**

Number of pages and appendix pages
91

The airline industry is a highly competitive area of business. Due to different factors, like advancements in technology, globalization, and deregulation of the industry, dense saturation by airline companies is observed. In the face of tight competition, different strategies are applied by the companies to attract customers, increase sales and generate profit. One of the strategies is to provide better service onboard: catering, shopping, entertainment, etc.

The entertainment aspect of onboard service provision is a special area of attention since it can alleviate the unpleasant experience of long travels in confined space. However, the traditional method of installation of personal IFE screens on the rear surface of airplane seats comes at significant cost, and some airlines refuse to use the option. Nonetheless, there is a possibility of multimedia provision even without IFE screens.

The thesis project work aimed to develop the system of offline in-flight entertainment (IFE), including both hardware and software, that would work on board an airplane and provide access to entertainment media for passengers through wi-fi network. The work also included the evaluation of such system's cost, that have been becoming popular and been installed by many airlines around the world. During the thesis project, full-stack work was done to develop a web application. Moreover, hardware was purchased and set-up for the application deployment. The final system was a fully operational offline IFE, that was providing access to multimedia through connection to its wi-fi network.

Important to note, that the pandemic of COVID-2019 virus, that drastically affected the global airline industry in 2020, was not included in consideration of this thesis project. The virus broke out at a time of thesis report finalization and therefore its severe negative impact was not additionally evaluated and provided in the theoretical section of the report.

Keywords
In-flight entertainment, IFE, wireless in-flight entertainment, software development, full-stack development, software project management, web development, airplane software, entertainment software, java, angular, javascript, spring-boot, offline systems, offline wi-fi network.

Table of contents

1 Introduction 1
2 Theoretical framework 3
2.1 The current situation in the aviation industry 3
2.1.1 Brief history, liberalization and its impact 3
2.1.2 Industry in numbers 6
2.2 Preference factors of air travellers and the position of IFE among them 10
2.2.1 Proper service provision is key in the competitive air market 11
2.2.2 Analysis of preference factors of travellers 13
2.2.3 IFE systems' importance for travellers 19
2.3 IFE systems – product analysis, industry and trends 20
2.3.1 Background history 21
2.3.2 Challenges of traditional IFE systems 22
2.3.3 Wireless IFE systems 24
2.3.4 Industry trends 26
2.4 Conclusion upon theoretical and industry analysis 28
3 Development plan 30
3.1 Solution architecture 31
3.2 System components 32
3.3 What system would perform 34
3.4 What system would not be able to provide 35
4 Empirical part 36
4.1 Development of the software 36
4.1.1 The building of Angular application 37
4.1.2 The building of Java application 52
4.1.3 Integration of Angular with Java 57
4.2 Integration of the software with the hardware 65
4.2.1 Server-side configuration 66
4.2.2 Configuration of wi-fi router 73
4.2.3 Deployment of web application on the server 77
4.3 Testing of the wIFE system 77
5 Evaluation and discussion 80
6 Thesis conclusion 82
7 Recommendations for further works 84
References 85

1 Introduction

The in-flight entertainment system, also known under its abbreviation as IFE, is a set of hardware and software, installed onboard the planes to provide access to multimedia for passengers and serve as a mean of entertainment during the flight. People who choose to travel by means of air transport often have to experience a trip in a confined space for long hours. While there are many ways to make the trip more enjoyable or easier to undergo for travellers, the presence of IFE can potentially benefit to the provision of a smoother flight by airlines to their customers. Passengers could use the IFE systems to pass the time in the air, and, depending on the system's content, be more satisfied with the services of an airline. Such outcome, in turn, could bring better ranking for the carrier and subsequently result in better sales. Assessment of the actual fierce competition in the global airline market could lead to an assumption that provision of entertainment for passengers during an aero trip, can be one of the factors leading to a better competitive advantage.

Traditional IFE systems are associated with flat screens presented on the rear side of passenger's seat, that often tell important information about a flight to travellers, and stream entertainment content. Although IFE systems can contribute to the overall service quality, the downside of the traditional systems is their cost, that comes in both upfront investment of purchasing and installation, and additional weight to an aircraft that results in more fuel consumption and respective higher expenses. While manufacturers of traditional IFE systems develop new solutions, that are more affordable and weighs less, airlines still have to make a decision between the system's costs and benefits that it provides. In fact, many airlines, in an attempt to save on costs, do not install such systems at all, while others consider mounting of such costly equipment on planes dedicated for long-haul flights only. Therefore, traditional IFE systems bring a cost-benefit dilemma to many air passenger carriers, that is not favourable for some industry members.

Modern technologies can come forward with a better solution, that costs significantly less and is immensely lighter. Increase of the affordability and popularity of smartphones and other personal gadgets with screens and support for wi-fi connectivity, as well as flight mode of operation, led to the creation of wireless in-flight entertainment systems – wIFE. They also consist of hardware and software, but unlike in traditional systems, wIFE relies on devices of passengers and additional equipment for the creation of a private wi-fi network. The system in overall is much lighter because it requires only one central storage and processing computer, a few wi-fi routers for network creation and devices, that are brought by passengers. Software part is essential for wIFE as well. It is needed to create

an interface for the user and feed the multimedia content. Wireless in-flight entertainment systems, therefore, can resolve the dilemma of traditional entertainment systems for airlines.

The thesis project aimed to develop such system similar to wIFE and explore the whole process of its development, and component infrastructure required. Hence, the work would have included the development of the software, obtainment of the needed hardware and overall set-up of the system. The thesis work was not intended to develop an actual wIFE product, that could be easily mounted into a plane. Instead, the goal was to develop a system with similar capabilities, hardware and software, that in total would create a resemblance to commercial wIFE products. The whole development part of the work would have significantly contributed to increased expertise in important ICT areas like software development, IT hardware, infrastructure, system configuration and integration.

The thesis work also included research into the airline passenger industry and overall IFE industry, with an observation of the current situation and trends. It also included research into the importance of in-flight entertainment systems and deeper analysis of their significance, benefits and costs for passengers and carriers. In other words, the research part had to determine whether IFE systems are important for passengers, and would airlines get a substantial benefit through acquiring of such systems. It also had to evaluate the situation on the market of entertainment hardware and software for aircrafts and reveal the favourable tendencies among the airlines for system type selection.

2 Theoretical framework

A thesis work related to the development of an in-flight entertainment system accessible by passengers through their smartphones and wi-fi network could hardly omit the subject of the importance of the IFE system for passengers and respective aviation carriers. Therefore, the first part of the report is dedicated to the analysis of the global aviation market, trends in the industry of IFE systems, and several research papers aimed at determining directly or indirectly the benefit such systems provide to passengers and adopters. Crucial to note that while there were many studies made to determine preferences of passengers, only a few were dedicating enough attention to IFE systems, and only one research paper was concentrating exactly on the importance of IFE systems for passengers. Nonetheless, enough material was collected and studied to make the aforementioned assessment.

2.1 The current situation in the aviation industry

The commercial aviation industry is almost one hundred years old and has been booming during recent years. The global situation in the skies can be represented by numerous airplanes of different airlines, that have been connecting more and more cities and carrying more and more people every year. The industry is profitable, and its positive performance is supported by different factors, like continuous liberalization of skies and borders, technological achievements, crude oil price situation, and overall improvement of the financial well-being of people around the globe. The drop of oil prices, the introduction of a new generation of aircrafts, and application of digital tools allowed to reduce costs and make aero travel more affordable for people. Cost reduction was also influenced by changes in local and international aviation regulations, that also eased running of the business, gave more freedom and opened new destination opportunities for airlines. The increasing population of the Earth and improving economic situation per se, coupled with continuous elimination of travel borders among countries, supplied an influx of customers who could and want to travel. Airlines have been soaring high like whenever before.

2.1.1 Brief history, liberalization and its impact

The industry is said to be established in the 1920s when the Air Mail Act (1925) and the Air Commerce Act (1926) were signed in the United States of America (Federal Aviation Administration, 2017). These were also years when new aircrafts appeared, that could be used in the commercial passenger market. However, the real development and boom of the industry started when countries one-by-one commenced adopting regulations that gave more freedom for airlines to run on their own. Further agreements on the

international level and liberalization of skies additionally spurred growth for companies, who were able to fly freely not only in internal but in external markets also (Gourdin, 2015).

The first country that provided more freedom to airlines for the domestic market was the USA. There, in 1978, the Airline Deregulation Act was passed and signed, which released companies from state ownership and gave them rights to operate on basis of competition rules of open market (Gourdin, 2015, p. 69). Other countries followed the tendency. For example, European Union deregulated its air through several legislative packages in the ’80s and ’90s of the twentieth century. The main breakthrough was achieved with the Third Package that came into power in 1993 (Gourdin, 2015, pp. 70-71). The impact of such measures by state authorities allowed airline companies to determine themselves the quality and variety of services’ provision, selection of destination cities, application of own pricing strategies, etc. Even though the laws concerned the internal market, the industry received a significant boost from such freedoms.

One of the main results of industry deregulation was the emergence of new airlines that followed a cardinally different business model. Such airlines significantly reduced the price of their fares by leaving in ticket’s price scope only the basic service of transportation from origin to destination. Other services, like meals and checked baggage, were removed from the fare and could be provided to customers for additional purchase. This was a no-frills business model, and airlines who followed it were called low-cost carriers – LCCs, or budget airlines. Airlines that continued provision of a full range of services, also known as full-service carriers – FSCs, did not consider budget airlines to pose a significant risk for their business (Gourdin, 2015, p. 27). However, LCCs were very attractive in the eyes of many travellers due to low prices and flexibility of services to choose and pay for. Therefore, the expansion of low-cost airlines was rapid, and by 2014 quarter of all travellers globally chose budget carriers for their trips (OECD, 2014, pp. 6-7). No-frills business model made air travel affordable for many people and allowed the less affluent populace to travel by air. Such a business model would not be permissible without deregulation actions initiated by many countries at the end of the twentieth century.

As it was mentioned above, market entry by low-cost carriers posed a significant competition to the existing full-service airlines. The key term is the competition that was established on the internal aviation market due to deregulation enforcements. Due to the competition companies started to apply different strategies to attract customers and stay profitable. Besides the introduction of budget airlines, other changes were observed also. Companies started to establish alliances, frequent flyer programmes and even

introduction of hybrid business models to tackle the competition with LCCs (OECD, 2014, pp. 17-19). All these measures were an indication of fierce competition and together with other factors resulted in positive outcomes for passengers in terms of pricing, flexibility of services and destination range. Arguably, open competition among airlines boosted competition among aircraft manufactures too, that in response have been investing a lot in the development of a new generation of aircrafts that could fly further, transport more weight and consume less fuel. A prime example for such statement could be a development of A350 series aircrafts by Airbus, in response to Boeing's Dreamliner and dissatisfaction by Airbus's main customers with initial response decision of modification of their existing A330 model (Pallini, 2020).

Figure 1. Airbus A350-1000 and A330-900neo (Airbus SE 2018)

Figure 2. Boeing 787 Dreamliner (The Boeing Company 2019)

Another action that has been impacting the industry is the external liberalization of the market and skies. Besides the measures that were applied by countries regarding aviation laws within their borders, many states have been establishing agreements among each

other on the international level that eased the process for airlines to fly over the air space of one country or arrive in a specific country. The best example could be U.S.-E.U. Open Skies Agreement which was signed in 2007. The agreement simplified mutual market access to airlines operating in USA and EU (European Commission, 2017). Countries in other parts of the world have also been coming to similar agreements. One of the most recent similar agreements came into force in Africa when 23 countries signed the Single African Air Transport Market in 2018 (Oxford Business Group, 2018). Such liberalization progress, together with the creation of new aircrafts mentioned before, brought further competition from the internal market to the international market also. For instance, the shared presence of budget airlines on long-haul trips was only 1% in 2014. However, by 2018 many of LCCs airlines had acquired a new generation fleet and therefore were able to carry 8% of passengers on the long-haul trip segment. Full-service carriers had to apply hybrid model measures in response to the increasing presence of budget airlines on long-distance trips. As a result, FSCs introduced affordable fares to customers with hand-baggage-only conditions (Hunt & Truong, 2019, p. 171). Again, this further stimulated competition and gave benefits to passengers in terms of carrier selection and service flexibility.

Actions started by governments in the late twentieth century, aimed at opening the skies to market had a significant effect on the commercial aviation industry. Companies were able to make their own strategic decisions and operate in response to the mechanisms and laws of the market. In the positive side for airlines, this resulted in more freedom and opportunity to conquer new niches. However, at the same time, it boosted competition in the industry. Further international agreements established regarding global skies had also an impact on industry expansion and competition strengthening. In overall, the positive impact was experience not by airlines only, but also by travellers, for whom air travel became more affordable, due to new business models and outcomes of competition.

2.1.2 Industry in numbers

The positive results of the impact of the deregulation and liberalization measures and overall performance of the commercial passenger aviation industry could be conveniently provided in numbers. The industry has been performing well for the last decades. It also faced a rapid improvement after the financial crisis of 2007-2008. While direct factors, like the aviation deregulations and advances in aircraft technology, were the important stimulants, still they were not the only. Other factors, like the price of crude oil, from which the fuel for aircrafts is made, and the improving economic situation of family households globally had an impact too.

At the moment, the airline passenger industry is booming and is saturated by many airlines worldwide that compete among each other for the choice of travellers (Oxford Business Group, 2018). Although the industry was hit hard during the financial crisis of 2007-2008, it gained a strong ground later significantly rose during the next years. For instance, during the first years of the crisis aftermath, many prominent investors were wary, cautions and even discouraging on the topic of commercial airlines. However, when the companies started to report positive profitability, those investors became one of the main shareholders of the industry (WNS Global Services, 2017). In general, airline industry entered the twenty-first century with negative profitability performance. In the USA alone, airlines had a cumulative loss of about 35 billion USD through the 2001-2005 period (McCabe, 2006). The value was steadily growing and going from a negative to positive scale, achieving 5 billion USD profit in 2006, which was almost tripled in the consecutive year.

Although it was the same year when the financial crisis started, the real impact of it was borne by the industry in the following years (IATA, 2015). The initial negative performance and subsequent hit by the crisis could be the reason why investors were wary and cautious regarding the industry. However, from 2012 and onwards, net profits accelerated to increase, and therefore the same cautious investors became one of the main shareholders of the industry (WNS Global Services, 2017). In 2012, according to the International Air Transport Association, global profits of airlines stood at a value above 5 billion USD. It took the next two years of stepped annual increment to exceed the performance record of 2007 before the financial crisis hit the world; in 2014, according to the same source, global profits of airlines stood above 16 billion USD (IATA, 2015). It was still less than what experts estimated by almost 20% (Cederholm, 2014). Nonetheless, the global aviation market made a cardinal improvement in terms of profitability in 2015, when airlines more than doubled their profitability and showed a value of 35.3 billion USD (IATA, 2016). The performance value since then had been above 30 billion thresholds, with a record of 38 billion USD in 2017 (IATA, 2017, 2018, 2019).

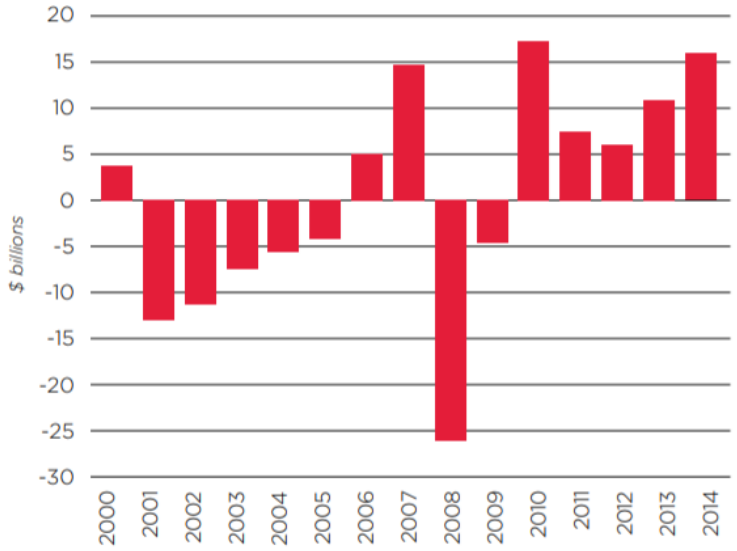


Figure 3. The airline industry's net profits (IATA 2015)

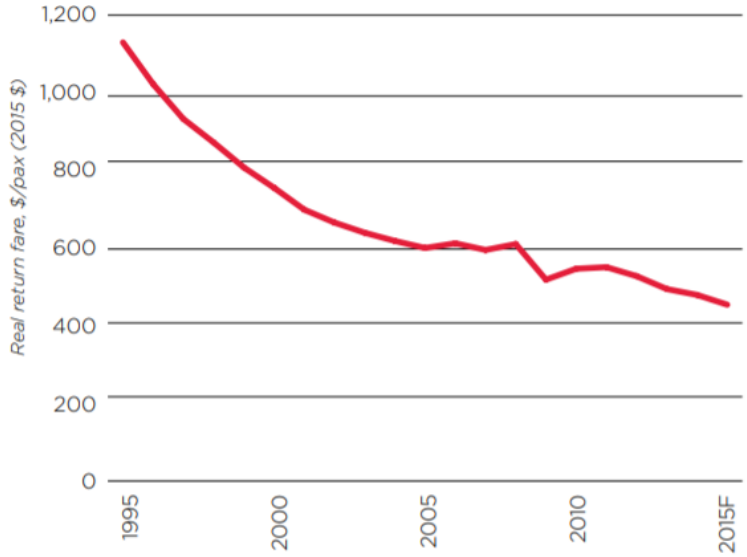


Figure 4. The global average real return fare (IATA 2015)

Such significant performance of the industry would not be achieved without boosting factors. As it was mentioned before, direct factors affecting the industry, like international and state laws and regulations, and the emergence of new technologies used in the production of civil aviation fleet, resulted in positive and competition stimulating outcomes for the market. The outcomes resulted in more flexible fares and a constant increase of air travel affordability for many travellers around the world. In fact, according to Kent N. Gourdin from the University of Tennessee, prices for airfares dropped by 50% in the period from 1995 till 2015 (2015, p. 57). IATA (2015), on the other hand, indicated that the global average price of a real return ticket, for the same period, decreased almost three times. Gourdin additionally stated that the reduction in air ticket prices would continue the

tendency at a rate of about 1-1.5% annually (2015, p. 86). These observations could be justified not only by the competition and technological advances but also by the situation with crude oil prices that directly resulted in cheaper prices of plane fuel, derived from the oil. As provided in plots from IATA, fuel prices have an almost direct relation to the global prices of oil (IATA, 2019), which was already observed to positively impact airlines in 2014, when the price of crude oil began to fall on the global market (Cederholm, 2014).

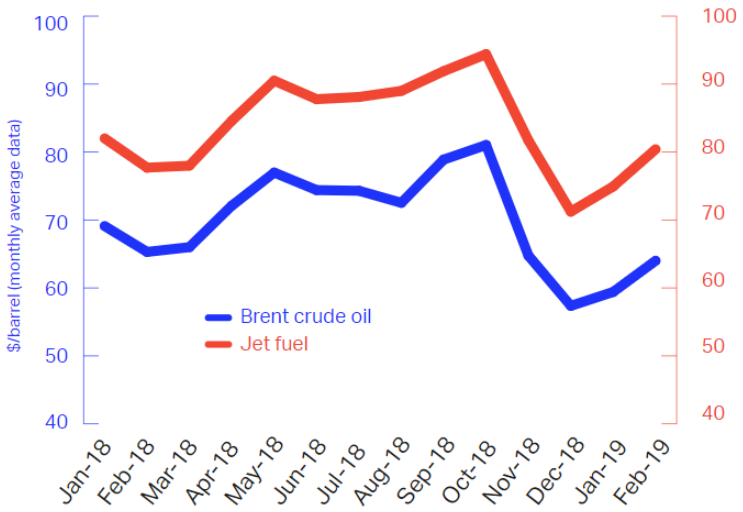


Figure 5. World oil and jet fuel prices (IATA 2019)

Another aspect of the dynamic world, that played a contribution to the airline industry was the improving economic situation of people globally. While the fares had been becoming more affordable each year, increasing financial wellbeing of people also resulted in a higher number of people who could and wish to travel, which resulted in more customers for airlines (Oxford Business Group, 2018). It is indicated by KMPG analysts that the annual passenger growth was indexed on average by 5.5% (Tozer-Pennington, 2019), while according to the Oxford Business Group, between 2012 and 2017 – years after a rapid recovery from the financial crisis and peak airline performance, the annual increment was measured at 6.2% (2018). In another passenger traffic improvement indication, it was shown that in 2018 the number of air travellers was 3 times higher than in 2000 (IATA, 2019). Forecasts from the experts of the industry envision the volume of passengers to reach 7.3 billion in 2034 (Gourdin, 2015, p. 85). In a cross-dependent manner, the number of city-pairs, i.e., the number of unique destinations increased also. Gourdin evaluated that between 1995 and 2015 the number of city-pairs almost doubled (2015, p. 57), while IATA indicated the actual doubling between 2000 and 2018 – from about eleven thousand pairs to twenty-two thousand pairs (2019). In overall other factors played yet another significant role in the positive development of the airline passenger industry. The industry, in turn, reacted in actions that continuously improve the accessibility and affordability of air

travel. In other words, a mutual connection exists between the airlines and one of its main stakeholders – travellers.

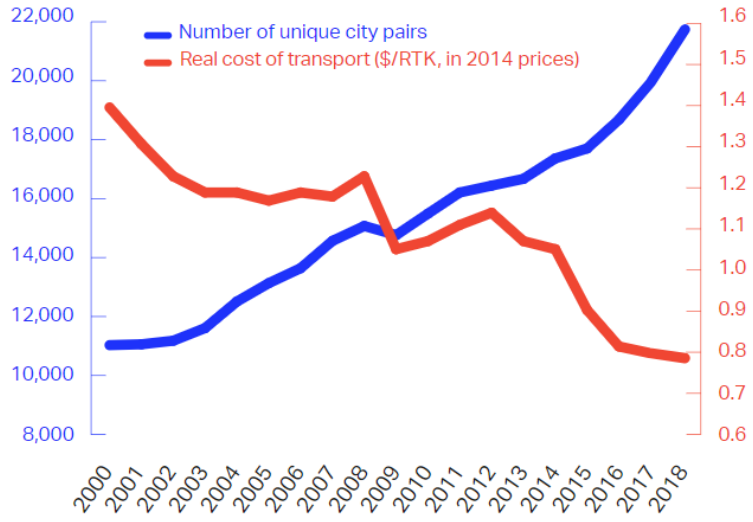


Figure 6. Unique city pairs and real transport costs (IATA 2019)

It could be concluded in the overall that the current situation on the aviation market has a positive dynamic. Due to the historical and continuous factors aimed at liberalization of the industry and international skies, civil aviation gained a boost and room for expansion and application of new business models. Additional factors of technological improvements, improvement of the global economic situation and recent drop of oil prices contributed to the soaring performance in the profitability of airlines. At the same time, however, companies have faced a tight competition in a wholly saturated market, where a win over travellers' choice is a guarantee of business's successful running.

2.2 Preference factors of air travellers and the position of IFE among them

Saturation by the presence of numerous companies and resulted competition on the commercial air passenger market brought airlines to concentration on the main target – attraction and retainment of travellers and customers for increased sales and hence better profits. Travellers, however, are different: they could have different reasons for travelling, originate from different cultures with different values, after all, be of different social status. Hence, airlines need to determine factors and services that are crucial in the eyes of potential customers. For instance, different business models, explored previously, were aimed at targeting a specific niche of passengers or conquer the attention of all travellers: low-cost carriers concentrated on light and budget travellers, while followers of hybrid model tried to balance between modest and affluent customers.

Nonetheless, some generalization of important factors for passengers could be made. More importantly, the significance of the in-flight entertainment systems has to be established. Such systems, again, provide access to different multimedia content, that could potentially ease the trip for people aboard airplanes. At the same time, however, IFE systems appear as an additional cost for airlines. They need to be purchased, installed and maintained – all of which can add investment and operational expenses to companies. Nonetheless, if the entertainment could appear to be a significant factor in preferences of travellers, airlines should consider making actions towards investing in entertainment on-board their flights.

To determine the factors and preferences of passengers, several papers were studied. The accent was made on the selection of research papers that either included the IFE factor or concentrated on it. Several works were obtained and analyzed, many of which were published aviation industry journals. Thesis papers regarding the subject of passenger preferences in the aviation industry were also found. Among the studied works, only one work directly aimed at determination of IFE as a success factor on the competitive airline market. However, materials obtained were enough to deduce conclusions regarding key on-board features and position of in-flight entertainment among them.

2.2.1 Proper service provision is key in the competitive air market

In commercial passenger aviation, airlines compete with each other through the provision of services for passengers. Companies, a priori, are involved in the service market, where the satisfaction of customers is a priority and is directly linked to profitability. According to Kent Gourdin, airlines should understand their customers, especially their needs and desires, and be able to provide services, that are aimed to satisfy customers' needs, and if possible be of quality, higher than expected by travellers (2015, pp. 95-96). In his work, Gourdin also provided the Service Quality model, that schematically represents the links and potential gaps between airlines and passengers. The model is provided in Figure 7.

In the model reflection aspects of air travel by passengers is accumulated into a factor of expected service, that according to Gourdin should be met and satisfied on required quality levels (2015, pp. 96-97). Different scenarios of poor need determination, provision or quality are represented by gaps of different levels. Gap 1 occurs, when companies make a wrong evaluation of the needs of travellers. The second level gap appears when customers are well-studied, but airlines are unable to provide the services, and therefore lose the satisfaction and number of passengers. The next level scenario in gap 3 is observed under poor delivery of services when the execution of service provision on the

place is diluted by ground or crew members of an airline. Such behaviour detriments the overall experience of a traveller, and negatively affects the company's image. Gap 4 is constituted by violated promises made by companies, yet not provisioned at all. Finally, the fifth gap appears when the experience was different from the expected quality. All five gaps can lead to customer dissatisfaction and eventual passenger capacity drop, which has the potential of a strong hit on the airline's profitability. Therefore, understanding of customers' needs and provision of the right services is crucial in the competitive airline world.

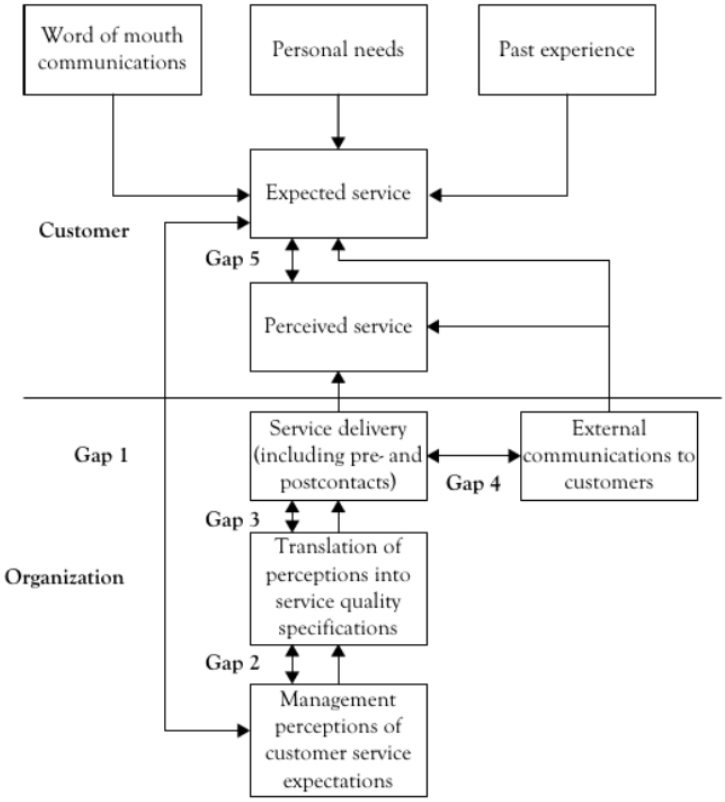


Figure 7. The Service Quality Model (Gourdin 2015, 95.)

The regard of customers attraction as one of the factors of success for airlines was also noted by Dr McCabe (2006). In his analysis article, McCabe used a key success factors framework applied to eight major airlines in the USA, selected by the share of sheer traffic volume. According to the professor, companies with proper management of the factors could perform at least two years without losses; hence came their success definition. The original count of the factors was twelve, but McCabe properly assumed the considerable global expansion of the airline industry and therefore noted about the fluctuation in the number of factors. Nonetheless, all factors aimed at improving airlines performance in four main areas, the first of which was customer attraction. In contrast to Gourdin's model, who separated the range and quality of services, McCabe stressed on the importance of the

overall service attractiveness, omitting specification of variety. However, a similar significance to the promotion of airlines' services was conveyed in McCabe's article. Providing a simplified formulation, for profitable operation, among other areas, airlines should outperform their competitors in the attraction of travellers, which is comprised by two out of twelve factors of success: service attractiveness and promotion strategy. Without the provision of the services that recall customer needs and without proper marketing of the airlines itself, companies arguable unlikely to sustain the competition.

Importance of service quality for airlines was also elaborated in the work of Korhonen (2019). Korhonen stated that the provision of quality services by airlines could contribute to better image creation of an airline, which in turn could lead to the accrue of loyal customers. Loyal customers, in turn, would become less demanding and less impacted by an occasional quality reduction in services. Nonetheless, it is stated that that provision of high-quality services is crucial. Moreover, Korhonen divided the process of service perception by travellers into two dimensions: technical and functional – where technical dimension was referred to the service product itself and its value, while functional dimension was related to how the service was delivered to a customer. Furthermore, the author derived determinants of service and quality in the airline industry. Korhonen stated that factors like appeal, accuracy, delivery by the company representatives and empathy are crucial in service creation and provision processes in the airline industry. Those deterministic factors can be used to measure the quality of services by airlines and help companies in the service creation process (2019, pp. 4-6).

2.2.2 Analysis of preference factors of travellers

The importance of proper service provision to air travellers is obvious. Airlines would benefit from faithful customers if the quality of services meets or exceeds customers' expectations. Therefore, and as it was mentioned before, it is the right action to determine the service products and preferences of the travellers; to recall, one of the gaps, provided in the service quality model of Gourdin, was concerned about the situation when companies were unable to determine the right product for the customers. Once more, works of different authors were used to analyze the preference factors of travellers, with the inclusion of in-flight entertainment systems.

Though there were several papers related to the analysis of air travellers' preferences and factors that could influence their choice, only one paper aimed at determining the importance of in-flight entertainment systems from the perspective of passengers. It was done in the UK, back in 1999, by Fariba Alamdari from Cranfield University. In her study, Alamdari wanted to determine whether in-flight entertainment systems, into which airlines

have been heavily investing in the ’90s of the twentieth century, were an influencing factor over the choice of an airline among passengers. The questions that the researched wanted to determine, among others, were the importance of IFE systems for passengers and the level appreciation of the systems by air travellers (1999, p. 203). In her research a total of one hundred people were surveyed; the travellers were subdivided onto two groups by the reason of travel: business and leisure. Both groups were asked to rate factors of air travel on a 0-5 scale, where 5 stood for the most important attribute (Figure 8).

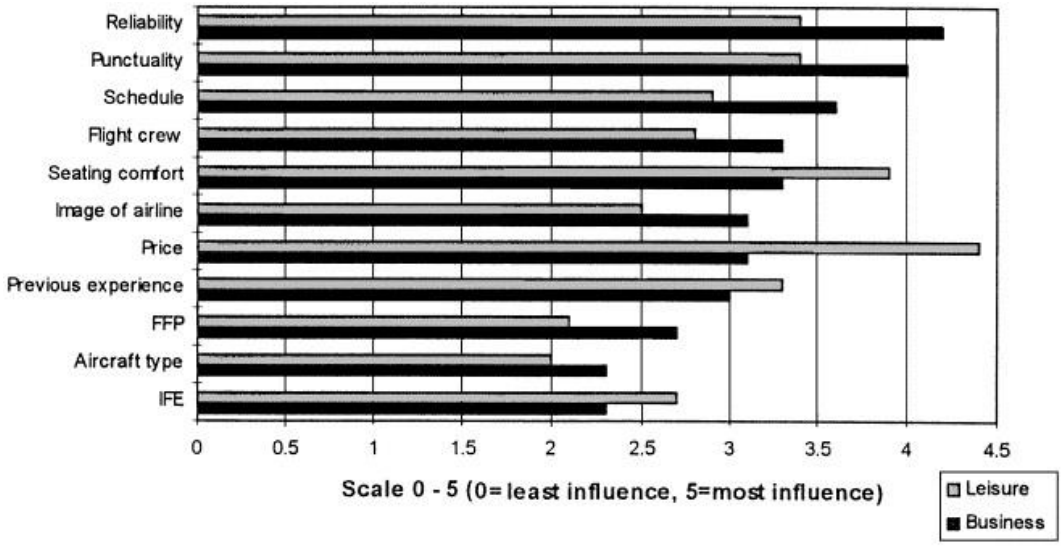


Figure 8. Factors influencing passengers’ choice of an airline (Alamdari 1999)

The results indicated that both groups did not consider IFE as an important factor. In fact, leisure travellers were more concerned about the price of a ticket, comfort of seats and reliability of their flights, along with punctuality. Business travellers, on the other hand, were prioritizing convenient schedules and attitude of the crew members, though they were also highly interested in the reliability, punctuality of their flights and comfort onboard the plane. Alamdari also noted, that her findings were similar to the results made earlier by SPAFAX Consulting, especially regarding the IFE factor (1999, pp. 204-205).

Nonetheless, the significant part of Alamdari’s work was an application of the second approach in her research, where the evaluation of IFE importance was not done through direct factor assessment by travellers. Instead, indirect questions were posed, that had to unveil the perception of the entertainment systems’ importance from the perspective of air travellers. It is important to note, however, that the second approach was limited to the scope of long-haul flights. The questions aimed to show what were the preferred activities among passengers during the flight, which source of entertainment they favoured the most, and what had improved in aircrafts in the past years in respect to the year when the

survey was conducted. The responders showed that entertainment was the second most preferred activity after relaxation and sleep. Among the preference of entertainment, movies outpaced greatly other activities like news reading and gaming. Finally, passengers reported that the most improvement in the aircraft was the IFE systems installed (1999, p. 206). In her conclusion, Alamdari stated that IFE was not among the main factors of attraction of customers. However, its provision could provide additional comfort to passengers, especially on long-haul flights and therefore could play a vital differentiation role, when a traveller is faced with options from different airlines that provide the same value of air travel (1999, p. 208).

Another study into the determination of important factors of air travel for passengers was done by researchers from the National Technical University Athens. In the study, eight hundred fifty-free people were surveyed in the Athens International airport – Eleftherios Venizelos, in December 2012 (Milioti, et al., 2015, p. 47). Researchers applied the multivariate probit model to analyze the data that was gathered through a sampling method of people at the airport. In other words, in addition to the initial assessment collection of air travel factor evaluation, the university members also tried to determine the significance of different factors in combination. Moreover, in their paper stress on socioeconomic and background of the air travel purposes was made; the paper aimed to reveal factors that had higher significance for passengers in processes of flight preparation, selection and ticket purchase (2015, p. 46).

Through the initial sampling, that involved evaluation of discrete factors, similar results to Alamdari's outcome, mentioned above, were obtained; travellers were more sensitive to the ticket cost, safety on-board, attitude of the crew, and convenience of the flight's departure and arrival date and time (Figure 9). Results for the IFE factor repeated and were one of the least important factors (2015, p. 49).

Factors that influence the choice of the airline carrier	Mean	St deviation	Min	Max
Fare (1 if it is an important factor; 0 otherwise)	0.882	0.323	0	1
Flight schedule (1 if it is an important factor; 0 otherwise)	0.587	0.493	0	1
Frequent flyer program (1 if it is an important factor; 0 otherwise)	0.216	0.412	0	1
Safety and reliability (1 if it is an important factor; 0 otherwise)	0.776	0.417	0	1
Connections until destination (1 if it is an important factor; 0 otherwise)	0.527	0.499	0	1
Large number of the cities served (1 if it is an important factor; 0 otherwise)	0.247	0.431	0	1
Friend's/Agent's recommendation (1 if it is an important factor; 0 otherwise)	0.145	0.352	0	1
Friendly and helpful staff in flight (1 if it is an important factor; 0 otherwise)	0.634	0.482	0	1
In-flight entertainment (1 if it is an important factor; 0 otherwise)	0.296	0.457	0	1
The whole airline's image (1 if it is an important factor; 0 otherwise)	0.722	0.448	0	1
Number of observations	**853**			

Figure 9. Evaluation of discrete factors (Milioti, et al. 2015)

However, the second part of the research involved the development of multivariate probit model and calculation of the coefficient of the factors combined with other aspects. While

the results still indicated low interest among passengers in Greece towards the in-flight entertainment, the calculation showed, that presence of IFE was more favourable in the situation of ticket cost increase (2015, p. 51). Travellers who were ready to pay more for their tickets, expected to receive, in turn, more services and better quality. Such an assumption was later supported in another model, where socioeconomic factors were put into consideration. IFE appreciation was higher among affluent travellers, who also purchase premium class fares (2015, pp. 50-51).

Another extensive research in preferences of air travellers was done by a group of researchers from Spain (Medina-Muñoz, et al., 2018). Researchers, who also profiled passengers by their socio-demographic status, were mainly driven by hypotheses that travellers with different profiles would have different expectations and needs for their flights (2018, p. 45). At first, over twenty research papers were analyzed to determine common twenty-eight factors that flyers find most attractive and important. The result of their initial analysis of the significant amount of works showed same results in terms of priorities: ticket prices, flight scheduling, cabin comfort and security of the travel (2018, pp. 46-47, 49). In-flight entertainment systems, though not mentioned directly, fell into a group of features that were least important to passengers according to the papers studied by the Spanish research group (2018, p. 49).

The subsequent part of the research involved surveying of travellers at an international airport in Spain; people were selected randomly and in total 406 completed questionnaires were obtained (2018, p. 51). In the survey, travellers were asked to rate twenty-eight factors of air travel that were obtained from the analysis of papers during the initial stage. Through the discrete sampling, it was again indicated that the price of the fare, scheduling, reliability and safety were the top factors crucial for customers of airlines (Figure 10). IFE was observed to be located third from the bottom of the rating (2018, p. 52).

Airline attributes	Mean	Standard Deviation
Safety and reliability of the company	6.65	0.95
Punctuality of flight departure and arrival	6.63	0.93
Convenience in making reservations and buying tickets	6.55	0.92
Direct flights, without stop-overs	6.51	1.17
Ticket price	6.46	1.08
Speed in ground services (check-in, boarding, luggage recovery…)	6.36	1.12
Making changes in the ticket without additional cost	6.35	1.42
Convenience regarding luggage (hand luggage and luggage to check)	6.28	1.22
Refund if not flying	6.28	1.47
Promotional prices	6.26	1.22
Ground crew (check-in and boarding crew) attention and service	6.22	1.26
Cabin crew attention and service (flight attendants)	6.15	1.35
Time scheduling of flights	6.13	1.39
Possibility of connecting with other flights	6.08	1.35
Frequency of flights (number of times per day, week…)	6.07	1.47
In-flight seat space	5.93	1.51
Past experience with the company	5.91	1.55
In-flight luggage space for hand luggage	5.84	1.42
Reputation/image of the company	5.82	1.44
Days of the week of flights	5.79	1.71
Price for checking additional luggage	5.61	1.85
Benefits of the fare class (seating comfort, in-flight mealsl/ snacks…)	5.12	1.80
Availability of in-flight Wifi	5.07	2.37
Free in-flight catering service (food and beverage)	5.01	1.84
Additional benefits from the frequent flyer programme	4.91	2.04
In-flight entertainment (music, magazines, radio/TV channels…)	4.71	1.97
Type of plane	4.66	2.06
Nationality of the company	3.83	2.25

Figure 10. Importance analysis of airline attributes, where 7 is the most important and 1 is the least important attribute (Milioti, et al. 2015)

Nonetheless, the results did provide the final part of the work. Additionally, statistical analysis of the data was performed to determine fluctuations in priorities based on the socio-demographic profile of individuals: gender, age, marital status, level of education, etc (2018, p. 52). Results showed that in particular regard to the in-flight entertainment, higher demand to IFE was originating from family groups, while single travellers, especially young, were least interested in such service (2018, pp. 53, 55). The authors, however, noted that the performed work and analysis were too broad in many factors and attribute of analysis, and at the same time relatively small in the number of samples obtained. Noting the satisfactory level of fit of the model in the statistical approach used, it was stated still, that general correlation with works of other researchers was derived (2018, pp. 55-56). Therefore, the paper had a relevant contribution to the determination of the passengers' preference factors.

The next paper studied for the analysis purposes of factors of significance among air travellers was written by two researchers from the USA. Jennifer Hunt and Dothang Truong (2019) compared the factor advantages of full-service carriers and low-cost carriers, that were involved in the transit of passengers over the Atlantic ocean; they

selected the long-haul segment of the airline market and wanted to determine reasons behind the emergence of budget carriers on the trans-Atlantic route (2019, p. 170). In their research, Hunt and Truong gathered opinions of 1412 travellers at the Los Angeles and Seattle-Tacoma International Airports (2019, p. 173). Prior to the survey, a literature review was also conducted, which provided initial factors of significance. Priority for the price, comfort (especially on long-haul routes) and scheduling were repeatedly stated. In-flight entertainment services were deemed unimportant. Moreover, quality of the service in general, through literature review, was revealed to have lesser significance for passengers (2019, pp. 172-173).

However, upon application of the analysis over the collected data from surveys, researchers provided a bit different rating. Airfare stood on its top position, being also the main factor among passengers for alignment with low-cost carriers, especially among the younger and uneducated population (2019, pp. 177-178, 180). The second priority on flights over the Atlantic ocean was seat comfort, which appeared to be a reason behind many passengers going with full-service carriers, even though the price rates could be higher (2019, p. 178). The reputation of an airline deemed to a factor of decision influence, which set a higher priority for in-flight service, that stood on the third place of factors' rating (2019, pp. 178-179). Direct dive into consideration of IFE was not made by the researchers, however, it was indicated indirectly that on long-haul flights IFE gains importance in combination with other services and aircraft features that represent the on-board service and seat comfort (2019, pp. 181-182).

Contribution of IFE systems towards the creation of higher value to passengers in the service provision was also regarded in the works of Finnish origin. Korhonen (2019), while stressing on the importance of the value creation through a high-quality and relevant range of services for passengers in the aviation industry, elaborated on the importance of the attainment of positive customer experience and indicated its relation to the positive image development of an airline. The experience with the service of airlines, according to the author, were divided into three groups: core, expanded, and experiences that fell beyond expectations (2019, p. 2).

The core services aimed at providing the basic need of air travel from an origin to the destination of passengers. Rely on the sole provision of basic needs would not be enough for the creation of high-level positive experiences for travellers. Therefore, it was stressed that services of expanded groups and services of the exceptional level played a considerable role in the building of impression, company image and attainment of competitive advantage on service market (2019, pp. 2-4). Since Korhonen indicated IFE to

be part of the expanded group of experiences, it could be stated that the entertainment systems should be acquired by airlines, for such systems are able to contribute to the expanded experience accrue.

Support for the importance of IFE in the provision of superior services by airlines was also traced in the research work of Ekaterina Tolpa from Aalto University (2012). In her Master's thesis work, Tolpa wanted to determine services customers of airlines pay most attention to; the similar purpose of the research as in other works referenced. Tolpa surveyed seventy-nine people, who all were maximum 35 years old (2012, pp. 48-49). The analysis made by Tolpa indicated that the ticket price and scheduling of flights had the highest priority for travellers, while onboard services, to which IFE systems belonged, had a low level of preference (2012, p. 53).

However, although the research found that the primary need for air passengers was to reach their destination, provision of IFE and other onboard services contributed to the superiority image of an airline by its customers, which in turn led to the development of loyalty attachment to the company (2012, pp. 53, 55). Tolpa also explored the improvement in customer satisfaction with air travel and the airline itself, when IFE was available as a source of entertainment; diversity and range of entertainment media distributed to passengers through IFE was also reported to have importance (2012, pp. 60, 63). Thus, IFE had an influence in combination with other onboard services to provide additional value to customers and improve the competitive advantage of an airline.

2.2.3 IFE systems' importance for travellers

Review of the literature indicated that in general, as a factor of choice influence, in-flight entertainment systems have very low appeal or importance compared to other factors. Customers of airlines are more concerned if they could save more costs on their air transit, get safely to their destination and more importantly arrive on specific date and time. The price factor was the most appealing to travellers' eyes in the considerable number of the papers reviewed; even in other sources, not mentioned above, price of the airfare was indicated to have the highest priority, which proved why low-cost business models managed to attract a vast amount of customers from the full-service airlines (McCabe, 2006). Therefore, provision of the in-flight entertainment system would not support the business run of airlines in the most favourable way.

Nonetheless, the studies also indicated that IFE systems play a considerable role as a differentiation factor and an attribute of over-expected value provision. Also, entertainment availability, or absence, is noticeable by passengers on long-haul flights; entertainment

systems on such routes provide additional comfort, activity option and help build a better image of an airline, that showed to support accrue of the loyal customer base. Provision of IFE, along with other amenity and onboard services, proved to be an attraction factor for FSCs from LCCs (Hunt & Truong, 2019). The systems also appeared to have an importance for family travellers and for affluent travellers who either preferred higher-class fares or aligned to the selection of non-budget carriers, from whom IFE was expected in the onboard service package. The conclusive statement about in-flight entertainment systems, in regards as a factor of preference of air travellers, would indicate their low position. But airlines should still invest onto provision of IFE since it can play a strong differential factor either in additional value provision or targeting of specific customer niche.

2.3 IFE systems – product analysis, industry and trends

As it became obvious through literature review, in-flight entertainment systems cannot promise a competitive advantage to airlines as a discrete factor. Instead, IFE systems come useful in combination with other onboard services provided by an airline, that in turn creates more value for passengers and thus contribute to the stance on a heavily saturated air market. Moreover, entertainment appeared to be important on long-haul flights and was highly appreciated by family and affluent travellers. Therefore, a suggestion for the installation of such systems could be made to the players of the commercial flight companies. Still, however, not all airlines provided entertainment systems onboard their fleet. Besides the relative contribution of IFE to the customer satisfaction, procurement of IFE systems can turn into a heavy investment, that could not necessarily yield any returns, and therefore many airlines omit the opportunity to fortify their existing services with entertainment programs.

Nonetheless, the manufacturing industry of IFE systems exists. Many companies, just like airlines, compete among each other to provide their solutions to be installed inside aircrafts. The original form of IFE screens on the rear side of passengers' seats is still present and included into hardware product of many modern aircrafts. Newer versions and models with different features and capabilities are developed by IFE companies worldwide. Additionally, wireless in-flight entertainment systems – the core objective of this thesis work, emerged in the 2010's and have been obtaining the attention of airlines.

In this section, the history of the IFE systems as a product, their technical nuances, diversity, development and trends will be described. It will be discussed why the traditional IFE systems are not always purchased and installed by airlines, and how do wireless entertainment systems stay out compared to the traditional seatback counterparts. The

section will also observe the situation on the IFE industry market and in what direction does it go. In general, it will be dedicated to the IFE product as a whole.

2.3.1 Background history

The entertainment onboard airplanes have been provided by commercial airlines since the origins of the industry. One of the oldest records, for example, that date back to the 1930s, state that passengers were entertained among other things with live music play and song performances by artists (Alamdari, 1999, p. 204). Other forms of entertainment, like newspapers and in-flight magazines, are still provided by many airlines globally. Mini-cinemas were also organized aboard the planes, and they were first introduced in the 1960's (Schawalder, 2014, p. 7). But only in the late 1980s, with technological improvements in TV-screens manufacturing, personal mini-screens appeared on the market.

The first personalized screens of IFE systems were introduced to passengers in 1988 and were a cabin privilege for premium class customers only. However, many airlines saw potential in the product and heavily invested in the acquisition of IFE hardware and software for their fleet, and even provision of the entertainment screens to all of their passengers. Throughout the 1990's global expenditure by air carriers into IFE systems grew more than four times. IFE systems of the twentieth century were representing the initial generation of the traditional IFE technology, that included hardware: visible screens, hidden wiring – and software. Such systems provided video and audio streaming capabilities, while some models included even communication systems with the land (Alamdari, 1999, pp. 204-205). Though robust, heavy and expensive, first-generation IFE were used by many airlines as a differentiating attribute, who dedicated special attention to the development of signature IFE portals: Singapore Airlines' KrisWorld (Singapore Airlines, 2020), Emirates' ICE (The Emirates Group, 2018) and Qatar Airways' Oryx One (Qatar Airways, 2020).

Throughout the first decade of the twenty-first century, traditional IFE systems were dominating the market. The industry aimed at perfecting the hardware and software aspects of them. Concentration was made on making them lighter, more affordable, user-friendly and interactive. Nonetheless, they were still considered to be a significant cost and therefore some airlines declined initiatives of IFE procurement or installed them on aircrafts, dedicated for long-haul flights only. Meanwhile, at the end of the first decade of the twenty-first century, and throughout the next ten years, advances were made in personal mobile phone technology; phones with wide screens and wi-fi network connection capabilities appeared in the late 2000s and had been becoming affordable to

people each year around the world (Schawalder, 2014, p. 8). Thus, new technological models became feasible – during the first years of the 2010's, wireless in-flight entertainment systems emerged on the market (Ramsey, 2011). In total, they were cheaper, much lighter, consisted of just few elements and relied on wi-fi compatible devices of passengers. The wireless systems were more affordable, and many airlines initialized provision of entertainment streaming via wi-fi networks onboard. In 2011 alone, more than 200 million passengers worldwide were provided access to entertainment during their flights (Shashank, 2013).

The traditional IFE products did not disappear with the development of their wireless counterparts. Backseat product market kept its operation in the second decade of the twentieth century. The decade was also a period for many airlines to update their fleet and acquire a new generation of aircrafts introduced in the previous decade not only by giants of Boeing and Airbus but also by smaller companies like Embraer and Bombardier, though later the giants purchased shares of their smaller competitors. The efficiency of new aircrafts and improvement of traditional IFE systems allowed many airlines to include interactive screens on seats of newly obtained planes. At the same time, some companies decided to install both types of IFE systems. The rate of adoption of IFE systems, however, had been different in different regions; the USA and Middle Eastern airlines had been investing more than their competitors in Europe (Schawalder, 2014, p. 17).

2.3.2 Challenges of traditional IFE systems

Traditional in-flight entertainment systems, that are mostly associated with a personal screen on the back of airplane seat, have been streaming entertainment content to flying travellers since the late '80s and early '90s of the twentieth century. As it was provided earlier in this work, entertainment has the potential of playing a differentiation role and provides more value to customers. Therefore, many airlines made significant investments in the installation of entertainment hardware and even the development of proprietary entertainment products. At the same time, however, many airlines refused installation of IFE systems from the beginning or changed to negative their regard towards the product. Such decisions could all be linked to the main disadvantage of the traditional IFE systems in terms of cost, that refrained airlines from an acquisition.

The first challenge of traditional IFE systems is the direct cost of purchase. Notably, at least in the 1990's, the purchase cost of IFE was only rising. According to Alamdari (1999, p. 203), at the dawn of the technology, the cost of a single unit for a seat stood at 1800 USD. Years later, in 1998, the price tag was indicated to be 6000 USD per seat, and at the same time, Alamdari noted, with reference to other sources, the estimation of an IFE

unit to be 10000 USD in the 2000s. In other words, while IFE was gaining popularity during the nineties, their unit price only grew. The price tendency estimation was correct, because according to Ramsey (2011) and later Schawalder (2014, p. 10), the cost of equipping each seat on a commercial passenger plane with IFE screens ranged between two and five million USD, which according to other estimations could represent 2% of the total cost of a single plane (Alamdari, 1999, p. 203).

The other disadvantage of the traditional IFE systems, linked to their upfront cost, is a negative return on investments. The beforecited Alamdari, in her research, found that although passengers were ready to pay more for their tickets to have access to the entertainment screens, the decree of the maximum permissible additional fare cost, coupled with a service period of IFE hardware of 3-5 years, could not lead even to the compensation of the investment (Alamdari, 1999, pp. 203, 207). Arguably, the work of Alamdari was published in 1999 and could only apply to the first generation of IFE systems. However, at least the stated service period of the IFE – 3-5 years, was also emphasized for systems of newer generation (Tsai, et al., 2014, pp. 69-70).

Traditional in-flight entertainment systems bring the additional load. The direct influence of load and fuel consumption is a fact; there is a well-known example from American Airlines that removed one olive from salad bowls served to passengers and thus saved 40000 USD annually (Tarry, 2019). The early generation of the traditional products was incrementing the total weight of a plane on average by six kilograms per IFE unit (Alamdari, 1999, p. 203). Although the newer generation of traditional IFE systems was lighter by minimum 30% (Ramsey, 2011), still, depending on the class level of the IFE hardware, just a display of IFE system was adding three to five kilograms of weight per seat (Tsai, et al., 2014, p. 69). Whereas the whole traditional system includes other hardware components, like wiring for instance (Alamdari, 1999). It was calculated, that each additional kilogram of weight on A330-300 – a wide-body long-haul aircraft from Airbus, resulted in fuel increase by 0.3 kg, which by the time of the study was equivalent to 1.1 USD (Tsai, et al., 2014, p. 69). Thus, the provision of multimedia entertainment to passengers during the flight resulted in additional cost through higher fuel consumption.

The downside of purchase and service of traditional IFE system hardware resulted in many airlines refraining from their installation: for low-cost carriers, the inclusion of IFE was violating the core idea behind the business model of cutting the costs and thus provision of cheaper fares; while for many full-service carriers, traditional IFE products were leading to more costs and less value for the airlines. Moreover, since airline industry had been responsible for 3% of total carbon-dioxide emissions, many governments

around the world had been implementing laws to reduce the impact of the industry on air pollution (Tsai, et al., 2014, pp. 65-66). Since the traditional systems revealed to add significant load, no wonder that many airlines removed the IFE screens from their planes and kept them on long-haul routes, albeit the investment history into the technology in the 1990s.

2.3.3 Wireless IFE systems

While the traditional IFE systems revealed to have a poor cost-value ratio for airlines' decision-makers, with the emergence of personal portable devices, that were able to support wi-fi connection and operation in flight mode, the possibility to apply different technology method in IFE concept appeared possible. Instead of installing monitors on each passenger seat, a wi-fi network, supported by few routers and main processing computer, could be configured – such system was named wireless in-flight entertainment system, or wIFE (Ku, 2019). The main computer would contain the actual entertainment content, it would also operate routers installed in the aircraft. The number of routers required could be five-six units per plane to provide a stable wi-fi network for all passenger on board the plane (GOGO LLC, 2016) (Taylor, 2019). The final element, which is highly important, is a personal handheld smart device, that could connect to the network, support flight mode and be able to stream the entertainment to passengers.

Wireless in-flight entertainment system products appeared at the end of the 2000's (Taylor, 2019), and gained a more stable position on the IFE market in the 2010s. The earliest record of wIFE implementation found dated back to 2010, mentioning Alaska Airlines signing a contract with GOGO LLC (Dow Jones & Company Inc, 2010). Alaskan Airlines was not the only airline to implement the solution, many other airlines did so too. In fact, in 2011 the number of wIFE installation was enough to supply over 200 million people with entertainment through their devices (Shashank, 2013). The growing tendency among passengers to bring their own devices aboard the planes and use them for entertainment purposes (Bachman, 2014; GOGO LLC, 2017, 2018; Kallonen, 2017, p. 19) further stimulated the popularity of wIFE systems and their adoption by airlines (Ramsey, 2011).

The most known manufacturers of wireless in-flight entertainment solutions are GOGO LLC, Kontrol S&T AG, Lufthansa Systems, and Bluebox Aviation Systems. GOGO LLC is a USA company and one of the leading players on wIFE manufacturing market. At the moment their flagship product is GoGo Vision, which since the product's launch in 2012 have been installed on more than 3000 aircrafts of company's customers like Qatar Airways, Delta, Japan Airlines and British Airways (GOGO LLC, 2017). Kontrol S&T AG is

a European player, who does not sell the whole wIFE product but supplies the hardware parts of the system, tailored specifically for wireless in-flight entertainment and connectivity purposes. Their product is ACE Flight 4600 and it was launched in 2012, since then it has been used by other vendors to complement their wIFE solutions (Taylor, 2019). One of the customers of Kontrol, and a market player itself, is Lufthansa Systems – another European representative. Lufthansa Systems was one of the first companies to develop wIFE solutions; the company started work on their project in 2008, and in 2011 managed to showcase their BoardConnect product at ITB Berlin (Taylor, 2019). Since then, their customers have been Condor, Virgin Australia, Eurowings, Azul Airlines and 13 more airlines. Finally, Bluebox Aviation Systems launched their product in 2014 (Taylor, 2019).

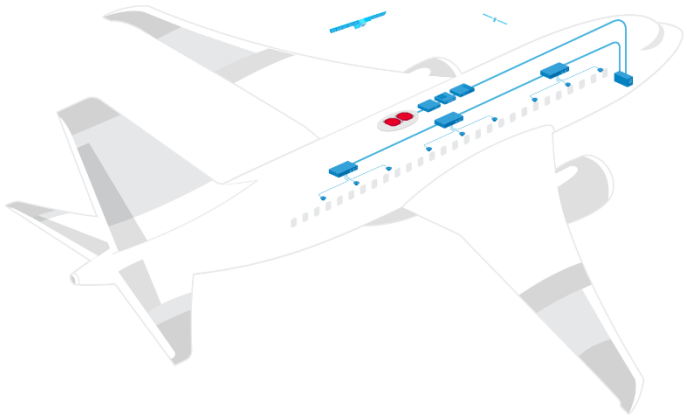


Figure 11. Schematic illustration of elements of GoGo Vision – a wIFE product (GOGO LLC 2016)

When wireless systems are compared to their traditional counterparts, the first advantage appears is a significant weight reduction. As it was mentioned before, one of the main disadvantages of the acquisition of personal screens for each passenger seat is weight load increment, that results in higher fuel consumption and therefore cost. At the same time, entertainment onboard yields to higher satisfaction among passengers, and therefore could play an important role for differentiation in a saturated market. That is why wireless IFE could appear as an optimal solution for airlines. For instance, GOGO LLC markets its solution to have a total weight of not heavier than a full standard beverage can (Bachman, 2014). Lufthansa Systems claims that installation of their wIFE product would

reduce the total weight of a plane by 80 tones[1] (Taylor, 2019). If fuel cost statement traced in the paper of Tsai et al (2014) is applied to the claim of Lufthansa Systems, then, at least BoardConnect wIFE product, would save over 80 thousand USD compared to the use of traditional backseat screens.

Another issue of traditional IFE systems was the upfront cost of purchase and installation. Particularly it was noted that the provision of seatback screens for each passenger on board the plane could cost between two and five million USD (Ramsey, 2011) (Schawalder, 2014, p. 10). Such an immense price tag is linked to the cost of hardware and its quantity. Traditional IFE systems consist of a mainframe unit and personal screens that connect to the main computer and feed multimedia content from it. Therefore, airlines are faced with a situation of a cost increment for each display mounting on a seat. The wireless solution, on the other hand, is much cheaper in terms of hardware cost. Indeed, the system requires a mainframe as well, but the wIFE does not require purchase and installation of display peripherals, instead, it relies on personal devices of the travellers and private wi-fi network created by several routers installed. Though it depends on the manufacturer's choice for the hardware: the main computer and routers - and sales price of the whole system product, that often includes software, still the final cost would be much cheaper for airlines. It is claimed that at least GOGO's solution is priced at 100000 USD, which is 20 to 50 times cheaper than the cost of traditional IFE systems (Schawalder, 2014, p. 10).

2.3.4 Industry trends

Comparison of the traditional and wireless in-flight entertainment systems revealed the advantage of the latest over the first one. Nonetheless, manufacturers of traditional IFE products, still operate and supply their customers with solutions to be installed in cabins of their customers' aircrafts. Airlines acknowledge the importance of IFE systems, but they require solutions that are lighter in weight, more compact, energy-efficient and cost less while providing more value and enjoyment to their customers (Ramsey, 2011). That is why manufacturers of traditional IFE systems have been investing in the creation of better products, that would tackle the common issues of their type of product. Panasonic Avionics Corporation developed eX3 and later NEXT series systems, that consumed less power by 25% and weighted by 39% less than previous generation series (Panasonic

[1] Such dramatic weight reduction was stated to be observed on Boeing 767-300 aircrafts, and strongly challenged the assumption of total mass load of traditional IFE systems indicated by Alamdari (1999) and Tsai et al (2014).

Avionics Corporation, 2019); Thales AVANT introduced I-5000 and I-8000 that also had a power weight measurements improved by 30% (Ramsey, 2011). Other traditional IFE manufactures, like Rockwell Collins and digEcor have also been making offerings on the market (Schawalder, 2014, p. 7).

Another reason why some airlines keep the seatback monitors could also be related to difficulties of the elder generation with modern technologies like smartphones and tablets. According to a report done by GOGO LLC, 47% of air travellers preferred to enjoy multimedia from the screens on seatbacks; such preference was mostly observed among people older than 40 years (GOGO LLC, 2018, pp. 11, 13). Airlines, knowing about the importance of IFE in additional value creation, prefer to keep the screens in their premium cabins. There was another tendency observed for traditional IFE systems kept in planes dedicated to long-haul trips (Ramsey, 2011). Besides the value that IFE provides in general for passengers during long-haul flights, it could be argued that seatback screens could entertain passengers in case their devices ran out of charge; in fact, high demand for power outlets among travellers was indicated in the research done by Schawalder (2014).

Nonetheless, the need to use wIFE has also been rising. Besides the advantages that wireless products have of the traditional mounted screens, the tendency among passengers in using personal devices has been increasing. Preference of utilizing own devices was indicated in research back in 2009 (Ramsey, 2011). Bachman (2014) in his article for Bloomberg Businessweek described how the affordability and spread of mobile devices were supported with incentives from airlines since the use of own handheld phones could have eased the strain from traditional IFE use. Preference for using personal devices was also observed in the research of preferences in the Asian market (Kallonen, 2017, p. 19). A study made by GOGO LLC revealed that 62% of air travellers used own smartphones, and when the preference of own device over mounted screens was raised, strong use of personal devices was observed only among 30%, while 47% of people did not have a specific preference which display to use (GOGO LLC, 2018, pp. 10-11). No wonder that several airlines have been investing in both types of entertainment systems: Delta, Turkish Airlines, Aeroflot, Singapore Airlines – these airlines have been installing both traditional IFE and wireless systems. It is expected that the trend to install both types of IFE systems would get more stable ground and popularity (Pierbon, 2019).

Another important trend that manufacturers of both types of IFE, and airlines themselves have been implementing is Internet connectivity. Importance of Internet access onboard the places has been stressed in many works studied during this thesis work. Kallonen

made a recommendation to include wi-fi connectivity to attract passengers from Asia (2017, p. 33). Internet connectivity grew its importance for both types of travellers: those who travel for tourism and vacation, and those who have a work-related purpose. If in 2014 only half of the travellers found connectivity worth of significance (Schawalder, 2014, p. 13), in 2018 the proportion grew to 78% (GOGO LLC, 2018, p. 3). In fact, 30% of people surveyed by GOGO LLC stated that they check if the Internet connection was provided onboard, prior ticket purchasing (GOGO LLC, 2018, p. 4).
So far, there were found two technological solutions to provide Internet access onboard planes. The first one involves the construction of land infrastructure, that would have capabilities of very broad bandwidth, up to the reach of planes in skies. Such a solution requires considerable investment into infrastructure, nonetheless, it provides a very stable network. GOGO LLC has built such infrastructure in the USA. That is why many domestic airlines in the United States included Internet connectivity during domestic flights; Internet connectivity was provided on one-third of all domestic flights in the USA in 2012. In Europe, however, the infrastructure is not present, because it consists of different countries and to perform a project of GOGO LLC type would be much harder. Nonetheless, there is another technological method: instead of using ground infrastructure, it is possible to use a satellite connection. Such solution is more expensive in terms of upfront cost, but it has been a favourable solution on international routes among many airlines, including US companies (Schawalder, 2014, pp. 10, 13-14, 17).

The final important trend, that is related to the growing demand for Internet connection on-a-fly and increasing affordability for airlines to provide such service, is the integration of the ground data about passengers on-ground and on-plane. Since the data is a crucial foundation for many tech giants to provide the right advertisement, airlines see a big potential of using IFE and information about their passengers to advertise services of their partners and thus obtain a new revenue channel (Ku, 2019). While the opportunity origins from airlines, the stress to implement it falls upon the IFE manufacturers; it was noted that airlines benefited a lot from new technologies, but they had been adopters and not definers (Tozer-Pennington, 2019, pp. 54-55).

2.4 Conclusion upon theoretical and industry analysis

Literature review of the materials related to the in-flight entertainment systems, their importance for airlines and passengers, and analysis of the technology's manufacturing market and trends led to the initial conclusions.

The first conclusion is that while as a sole service offering, IFE deemed to be insignificant to passengers and had no importance during the airline selection process by travellers,

the entertainment content was crucial in the creation of additional value to passengers, that could exceed their expectations and lead to higher satisfaction. IFE also plaid a considerable role as a differentiation factor, that could become highly important when a ticket buyer is faced with different offers, that satisfy the core needs of the traveller.

Airlines do not necessarily have to heavily invest in the purchase of the traditional systems, that are costly and lead to additional increased fuel expenses. Due to the rising affordability and spread of mobile devices, it is preferable to install wireless solutions, that are cheaper to purchase and service. Although many airlines still purchase seatback screens for their long-haul fleet, a scenario of wIFE provision and power outlets supply in each seat is also favourable. It is cost-efficient and could manage the situation of charge run-out of a smart device. However, to keep a high level of service provision and comfort for premium passengers, fitted solutions could be kept in business- and first-class cabins.

The final statement regarding the first part of the thesis report is related to the investment in Internet connectivity onboard. While it is not directly related to the IFE systems, that entertain passengers in an offline mode, due to the observable indicators of the rising demand for the access to the world wide web in the skies from both tourists and business-related travellers, airlines should invest more into the provision of such service. Especially European airlines, who showed less endeavour in that direction, should concentrate more on the matter. Stable and affordable Internet connection would further release airlines from the cost of entertainment provision since passengers would be able to access their favourite entertainment sites on the web. Moreover, airlines could use the onboard Internet connection as a channel of additional revenue streamline through advertisement or data collection.

3 Development plan

The development of an actual wireless IFE product, ready to be offered on aviation market, was not feasible due to several reasons: time constraint, budget constraint, human resource limit, specifics of hardware design. While the first three reasons mentioned were directly related to the fact that the whole development was a thesis work of a student, who developed the idea personally, the hardware design was perhaps the main reason. Actual commercial products can be well integrated into the cabin an airplane, so that decorative consistency of interior design is not violated. Besides, different planes have different internal wall and ceiling forms. Therefore, a market-ready solution would have first included proper cooperation and partnership with airplane manufactures or aircraft interior design companies, so that the few hardware elements, comprising the system, could be well mounted and fit into the whole body of any type of plane. Adding up the factor of cost and time limit also resulted in the inability to design a fully integrable hardware aspect of the wIFE. However, it was still possible to obtain commercially-available and affordable hardware, that could not necessarily be installed and fixed on a plane, but that could deliver the functional aspect of the product. Therefore, the practical part of the work was pruned down from the importance of hardware integrity with an airplane.

The software aspect of the project had also limitations applied to it due to similar reasons. The actual modern IFE solutions stream different type of entertainment content: movies, music, books, live TV. Many of them have integration with in-flight shopping and assistance services. They could also include authorization services, that if coupled with other software and hardware, could provide different range and quality of entertainment: for the economy and business passengers, for those who paid and did not. Mentioning the shopping and ordering of additional services, some airlines provide payment possibility through their wIFE. Implementation of such a feature would require significant development of security and payment transaction capabilities of the software. Reminding about the project limitations, it was instead concluded to develop software that would only deliver the video content to users; passenger authorization, payment support and other functionality were excluded from the project work.

Another decisive action related to the software part of the development was a selection of an interface type. There were two options: a web application interface and a stand-alone application interface. Although many airlines have a stand-alone application interface developed: Aeroflot, Turkish Airlines – development of such a system was not favourable. It would have required creation of a mobile application, that users had to install on their

devices prior to the boarding. Since there are at least two known mobile operating systems, that dominate the mobile industry – iOS and Android, stand-alone development for mobile devices would have included the importance of compatibility for those two systems, which translated to increased time consumption. Moreover, even if a native-friendly framework was used, like React Native, to tackle the platform issue, still to upload the software, at least for iOS devices, a fee had to be paid, which violated the budget constraint of the project. Additionally, wIFE access through a mobile application would have excluded users with laptops, who would not be able to install a mobile app onto their computers.

Web applications, on the other hand, could be accessed from any device that has web browsers; all modern smartphones, tablets and laptops have support for native and third-party Internet browsers. At the same time, web application development for this project would have incurred no direct of afterwards costs. Assuming also the claim that web development is faster and easier than stand-alone development, and the experience of the author in web development, it was decided to develop a web application interface. Such an application would be able to deliver content to the user independently of device used, as long as the devices support web browsing.

Testing in actual environments is an essential part of product development. However, noting the specific nature of a product to develop, actual tests had to be performed in skies onboard an airplane. However, during the thesis work, it was not possible to perform such tests. Nonetheless, it was possible to do tests on land, by turning the flight mode of operation on a device. Therefore, although the important part of the practical work – testing, was not in the skies, it was still planned.

In summary, the development plan involved the creation of a system similar to actual wireless in-flight entertainment products, commercially available on the aviation market. The hardware part of the system was freed from the interior integration importance of a plane. Budget constraint set to use and purchase of any affordable hardware that could support the system operation and access to it. The software part of the system had to be accessed via a web interface and it was limited to the provision of video content only. Other functionality features were excluded from the scope of the target product. Finally, tests of the system had to be performed on land.

3.1 Solution architecture

The essential components of even a commercial wireless IFE product are not that big in number. The schematic representation of the GoGo Vision solution included one main

computer, that stored the digital content, served as deployment and run environment for software, and served a set of routers. It should be added that the GoGo Vision product included also Internet connectivity through connection to a very strong proprietary infrastructure of GOGO LLC. Nonetheless, the Internet connection is not a compulsory functionality for IFE systems in general. Therefore, the hardware part of the desired solution should have also included one main computer and a set of routers. However, since actual tests onboard a plane were not planned, and a priori were not feasible, one wi-fi router would have been enough. The main computer would work as a storage and deployment unit, while the router would create a private network.

The main aspect of the software part of the solution had to be a web application, that would run on the main computer and be accessible via web browsers under connection to the wi-fi network. The application had to serve the multimedia content in a convenient and user-friendly way through its interface. The application had to also be able to access and read media files stored on the main processing unit.
Thus, the abstract system architecture would consist of a computer, wi-fi router and web application. The computer would store the media files, work as a deployment unit and serve the wi-fi router. The router would create a private network, available for connection to passengers via their mobile devices. Connection to the network would provide access to the entertainment content stored on the computer and fed via the running web application on it, through the application's web interface. In overall, the architecture would look like in **Error! Reference source not found.**.

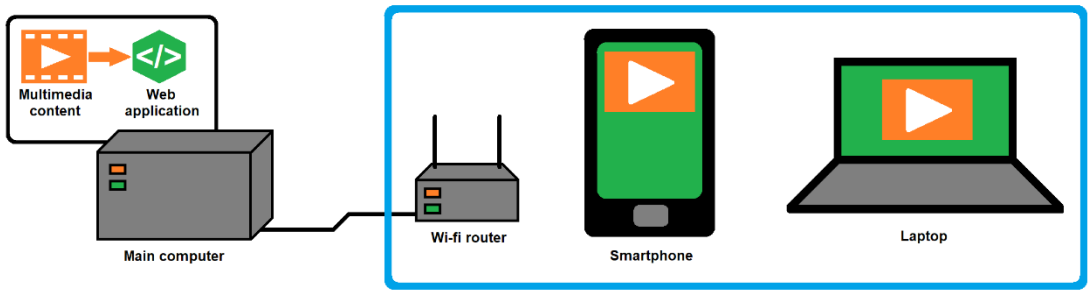


Figure 12. Schematic representation of the working wIFE system

3.2 System components

At the time of the development of the wireless IFE system, there were four devices in possession: two laptops that ran on Windows 8 and Windows 10 operating systems, and two smartphones of different brands that both, however, ran on Android operating system. Instead of purchasing a separate micro-computer for main functionalities, it was possible to use one of the laptops, while the rest of the devices could have been used during the

tests of the system. However, a scenario of a separate micro-computer was still considered. Often, micro-computers have no operating system installed. Therefore, such devices require an operating system to be installed; for example, an OS from Linux family of software.

Linux-based operating systems are available under different names, are mostly open-sources (though commercial versions are also available) and can be easily obtained and installed. It was decided to recreate the condition of using a Linux operating system on the main computer. The choice of the software fell for CentOS6 because that OS had been used and studied before by the author in other projects. However, to run CentOS6 on a Windows machine another software was required – namely a virtualization application. The choice fell for the VirtualBox from Oracle, that was openly available on the Internet. Thus, the main computer had to be a Linux machine, virtualized via VirtualBox on a Windows 10 laptop. The laptop was also used for software development purposes.

The important hardware component that was missing, was a wi-fi router. In order to manage the situation, an ASUS router was purchased in a local store of electronics. The selection was mainly influenced by the price, though the technical characteristics were highly suitable to the use in the project; in fact, some of its main technical features were similar to hardware of Kontrol S&T AG.

The web application had to be a full-stack software project, thus included both backend application part and the frontend application part. The backend was decided to be built with Java programming language, using the Spring Boot framework. Spring Boot framework was widely covered during studies of Software Engineering at Haaga-Helia University of Applied Sciences, therefore selection for the development language and framework fell due to the skills and knowledge assessment of the author. Spring Boot framework is based on the original Spring Framework yet provide more flexibility and acceleration in the development due to the almost complete abdication from XML configuration of a project in the standard Spring framework. Moreover, Spring Boot framework included an integrated Tomcat server, that freed from the need of web application server setup in the deployment environment: built files of Spring Boot framework are executable in any Java-provided environment.

Although it was possible to develop a full-stack project with Spring Boot entirely, it was decided to use Angular – a framework of JavaScript, for the creation of the frontend part of the application, i.e. user interface. Reasons for the selection of the Angular framework

were the flexibility of the JavaScript in web programming, ease of development in Angular (especially due to constitutional similarities with Spring Boot framework: component and service classes, dependency injection, etc.), and personal interest of the author in the Angular framework; thesis work provided an opportunity to better understand, learn and master the Angular framework through creation of actual web software.

The main computer mentioned in the hardware paragraph had to perform as a deployment environment for the web application and store the media content. Although it is common to use databases to manage and store data, due to the use of multimedia files, it was decided to implement a file management method and let the backend of the application to directly open and obtain data from the folders used for multimedia organization, stored on the computer. The main computer also required a Java Development Kit to serve the web application as the deployment and run environment. OpenJDK was a very suitable choice since it was an open-source version of Java than Oracle or IBM.

The complete component picture had the following structure and content. The wireless IFE system consisted of two main hardware items: a wi-fi router and a laptop. The laptop was used to develop a full-stack web application, which had a backend written in Java using Spring Boot framework and frontend written in JavaScript using Angular framework. The laptop itself operated on Windows 10 OS, whereas the plan was to use a Linux machine. Therefore, the work computer had VirtualBox installed, to recreate a CentOS6 machine, which had to be used as the main computer as described in the architecture of wIFE. The machine had to use an OpenJDK to be able to execute the web application, but it was decided to skip implementation of multimedia data management with a database, instead multimedia had to be accessed by the web application directly from folder systems. Additional third-party software was also used in the development, configuration and deployment process of the project. However, their mentioning is not important for they had no importance in the actual operation of the developed system.

3.3 What system would perform

It is important to mention again the extent and range of capabilities of the desired wireless in-flight entertainment system, planned for the thesis project:

- The system would create a private wi-fi network;
- Upon connection to the network from a personal device, an entertainment portal would be reachable via a web browser;
- The portal would have a friendly user interface, that would support both English and Finnish languages;

- The only available entertainment content to be fed would be movie trailers because actual use of movies would violate copyright laws;
- Theoretical passengers could use smartphones, laptops and other devices, that support wi-fi connection and have a web browser, to access the IFE portal;

3.4 What system would not be able to provide

At the same time, that planned wIFE system would not provide all capabilities of its actual commercial counterparts and would lack in the following range of features:

- The system would not provide access to the Internet. While browsers would reach to the portal, other websites would be inaccessible due to the absence of connectivity to the world-wide-web;
- The system would not provide authorization features for separation of economy and premium class passengers;
- Users would also be unable to make purchases via the application;
- Other types of entertainment like music, books and live TV would be also out of range of the system and its web application;
- The system would not provide flight status and flight-related information;

4 Empirical part

The practical and the main part of the thesis work involved the development of a system, consisting of hardware and software, that would copy the functionality of existing wireless in-flight entertainment products available on the aviation market. The goal was to write software, collect and configure the necessary environment, so that resulting system would allow a theoretical passenger on board a plane get access to the entertainment media through a handheld personal device or a laptop.

The practical development of a wIFE system had limitations in goals of a final product due to the time, budget and environment constraints. Such attributes of actual product development, and particularly for in-flight entertainment, like system testing inside a plane or cabin-mountable design of hardware were out of the scope of the project. The goal was to achieve the functional similarity of a system to the existing commercial products. However, due to the time constraint, even that factor had its limits.

The final product consisted of hardware items that were freely available on the consumer market of electronics in Finland. Except for the open-source operating system, the actual software for delivering content to personal devices of users was developed by the author. Software development part was the most time-consuming stage of the project. Though the system was well operating and providing access to the entertainment content through personal devices, there was still a lot of room for improvement of the developed IFE. Nonetheless, the developed system was working and achieved the set standards for the thesis project work.

4.1 Development of the software

One of the main components of the system was a web application that would run on the main computer and provide an interface to the entertainment portal of the wireless IFE to users. The application had to be a full-stack web software, and it was decided to write the backend part of it in Java using Spring Boot framework, while the frontend was dedicated for implementation to JavaScript and its Angular framework, that is based on TypeScript and thus resembles common paradigms of object-oriented programming and Spring Boot framework.

For the management and version control of both backend and frontend parts of the application, Git was used. Although the project was not intended to be developed in a team, and repositories were located only locally up until the project was finished, Git was still more favourable due to its flexibility, especially branching, and excellent possible

strategies of reverting mistakes and reverting steps in time: interactive rebasing, resets, reference logging, stashing. Other version control systems, like Mercurial and Apache Subversion, were simply too rigid and inconvenient for use in the project.

An interesting technique of integration of a Spring Boot project with an Angular project was discovered and initially applied to develop a single deployable file. However, as the project grew in size and complexity, the original method revealed to be inefficient and prone to build failures. Therefore, a new technique was developed to integrate the backend with the frontend, that performed successfully and allowed to produce just one executable file for the deployment. However, it is still important to note that while deployed as a single object, communication between the Angular frontend and Spring Boot backend was performed using representational state transfer application programming interface - REST API.

Result of software development of the practical part of the thesis work was a responsive application, that provided user-friendly interface on devices of any screen size and supported localization in English and Finnish languages. The application was able to obtain requested movie trailer files and provide them for watching to a user through its interface in a web browser.

4.1.1 The building of Angular application

The work on the web application development started with the frontend part of the software. However, it is crucial to note, that in the later stage of development, work was switched to backend and parallel work to integrate the use of REST API endpoints and apply proper models and interfaces on UI side, respective to the response structure from the backend. During the standalone development of the frontend, a mock of movie data was used. However, at the final stage, the data was obtained through HTTP calls to the backend, which in turn was reading requested files directly from the machine where the whole project was deployed, i.e. CentOS6.

For the development of an application’s user interface in Angular several requirements had to be met and satisfied. Angular required NodeJS of version 8 and higher (w3resource, 2020); the latest version of it recorded, after the installation and update, was 10.15.1. Node Package Manager (npm) was also a crucial player in the development because it allowed to import and use different libraries and improvement packages from the Internet, for example, Bootstrap and Ng-Bootstrap. Fortunately, the npm comes with the NodeJS and did not require a separate installation. Another element, which is not essentially required to develop applications in Angular, but significantly simplifies the

process, was Angular command-line interface (CLI). Upon the installation, the version used was 8.0.6 (Figure 13). The CLI provides a set of commands that accelerate the creation of essential building elements of angular applications and simplifies the process of testing, running and building of Angular applications.

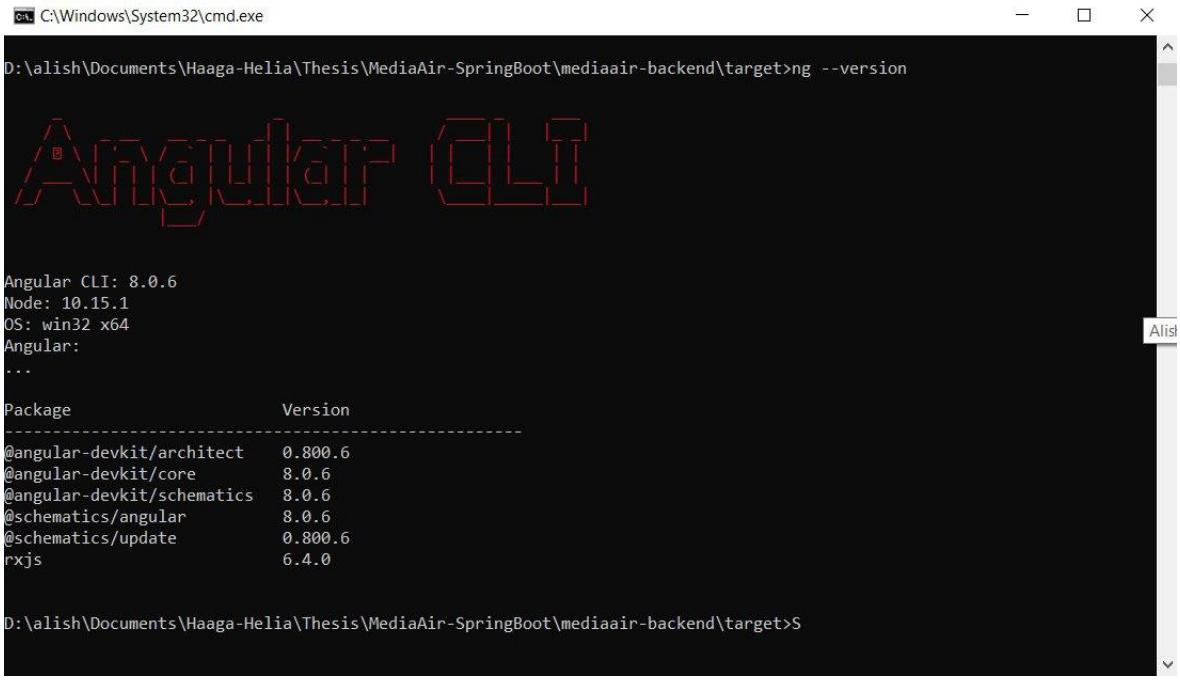


Figure 13. Version display of Angular CLI and NodeJS

Angular essentials

The paradigm of the Angular frameworks lies around three most essential parts: components, modules and services. Components are essential building blocks in the Angular framework. They are represented by ES2015 modules that are decorated with @Component directive. Each component provides both logic and template aspect of a specific view of an application. Components can be nested and in fact, Angular applications are trees of nested components (Figure 14). The runtime behaviour of components can be exploited with different life-cycle hooks (Google, 2015). For instance, specific data can be fetched at the initialization stage of a component (OnInit), or specific signal can be sent at the time when the component is destroyed (OnDestroy). Each component in Angular has a selector name, HTML template and style (the last two can be either inline or specified separately in other files). Angular apps always have the main root component, typically called AppComponent (Figure 15).

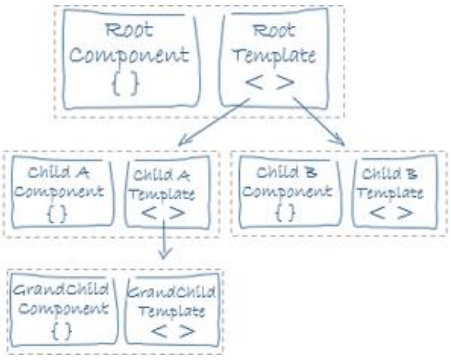


Figure 14. Component tree in Angular projects (Google, 2015)

```
src > app > TS app.component.ts > AppComponent
1   import { Component, OnInit } from '@angular/core';
2   import { BehaviorSubject } from 'rxjs';
3   import { TranslateService } from './translate.service';
4
5   @Component({
6     selector: 'app-root',
7     templateUrl: './app.component.html',
8     styleUrls: ['./app.component.css']
9   })
10  export class AppComponent implements OnInit {
11    uiLabels$: BehaviorSubject<any>;
12    isCollapsed = true;
13
14    constructor(private translateService: TranslateService) {}
15
16    ngOnInit() {
17      this.uiLabels$ = this.translateService.uiLabels;
18    }
19
20    translate(to: string) {
21      this.translateService.translate(to);
22    }
23
24    toggleCollapsed(): void {
25      this.isCollapsed = !this.isCollapsed;
26    }
27  }
28
```

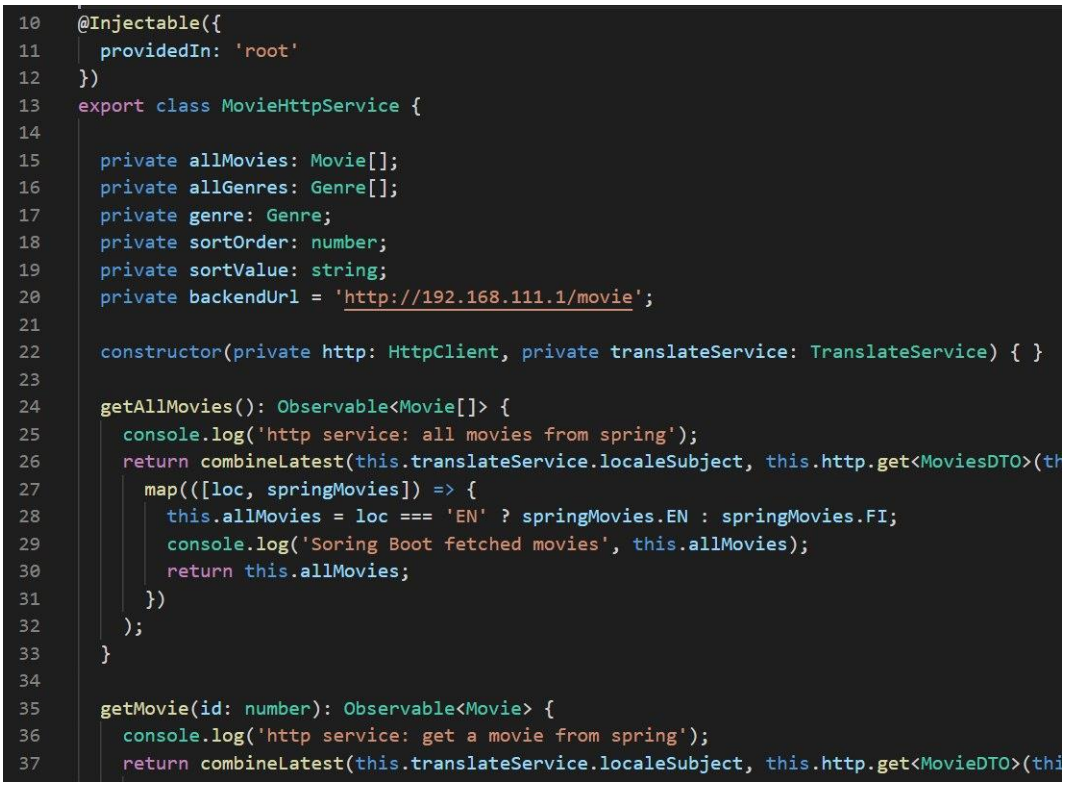
Figure 15. AppComponent's content in the frontend part of the project

To become visible for an Angular application, components must be registered through declaration in modules. Modules are another essential element of Angular application. They are also ES2015 modules themselves, that are decorated with @NgModule directive. They can declare components that belong to them, that helps in the organization of projects' structure. They can import other modules, to let their components and services to have access to features, that other components export. Angular provides a significant range of ready-made modules, for example, import of HttpClientModule from @angular/common/http would provide services of HttpClient for REST API calls. Since an Angular application must always have a root component, and components must always be declared, any Angular application always has a root module, that is also typically named

AppModule (Google, 2015). The AppModule of the developed application can be observed in Figure 16 below, pay attention that providers array is empty.

Figure 16. AppModule structure of the frontend

As it was mentioned, Angular modules can specify services that they provide. Services are another essential element of Angular framework. They are also EcmaScript modules and are distinguished by @Ingectable decorator. It is important to note, that in latest versions of Angular, a need to record module services in providers array became deprecated. Instead, it is suggested to set global access to a service through root specification in the “providedIn” property of services’ decorator object. For instance, the MovieHttpService had a global provision, but instead of writing it in the providers array of the AppModule, it was set to root availability through its decorator object’s property definition (Figure 16 and Figure 17). As it may have become clear, Services provide logical support for common actions across components of Angular application. For a Component to access methods of a Service, a dependency injection must be performed. Dependency injection in Angular is performed through property class creation of the Service type in the constructor of the Component. An example is observed in Figure 14, where AppComponent has a TranslateService injected into it. However, injection is not possible if either a Service or a Component were not provided or declared appropriately beforehand.



```
@Injectable({
  providedIn: 'root'
})
export class MovieHttpService {

  private allMovies: Movie[];
  private allGenres: Genre[];
  private genre: Genre;
  private sortOrder: number;
  private sortValue: string;
  private backendUrl = 'http://192.168.111.1/movie';

  constructor(private http: HttpClient, private translateService: TranslateService) { }

  getAllMovies(): Observable<Movie[]> {
    console.log('http service: all movies from spring');
    return combineLatest(this.translateService.localeSubject, this.http.get<MoviesDTO>(th
      map(([loc, springMovies]) => {
        this.allMovies = loc === 'EN' ? springMovies.EN : springMovies.FI;
        console.log('Soring Boot fetched movies', this.allMovies);
        return this.allMovies;
      })
    );
  }

  getMovie(id: number): Observable<Movie> {
    console.log('http service: get a movie from spring');
    return combineLatest(this.translateService.localeSubject, this.http.get<MovieDTO>(thi
```

Figure 17. MovieHttpService at its final form

Another important recall about Angular is that its template structure is a tree of nested components, and therefore could create a false perception of potential navigation issue in a web application. Instead, however, Angular has very efficient navigation support among components through Router Modules. RouterModule has essential methods like forRoot – for parent root navigation, and forChild – for nested navigation. The methods should be fed an array of Route objects that should essentially specify which component to show upon a request to a specific path. It is possible either to directly specify the RouterModule in an app's root or section module, or wrap the router in a module itself and import the named module to a module required: in the thesis project's frontend application the RouterModule was wrapped in own AppRoutingModule, that was exporting the RouterModule and imported by the AppModule (Figure 16 and Figure 18).

src > app > TS app-routing.module.ts > ...

```ts
import { NgModule } from '@angular/core';
import { Routes, RouterModule } from '@angular/router';
import { MovieListComponent } from './modules/movies/movie
import { SeriesListComponent } from './modules/series/seri

const routes: Routes = [
  {path: 'movies', component: MovieListComponent},
  {path: 'series', component: SeriesListComponent},
  {path: '', redirectTo: '/movies', pathMatch: 'full'}
];

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }

```

Figure 18. AppRoutingModule that implemented navigation in Angular frontend

The final useful feature of the Angular framework is its base on the TypeScript. TypeScript appeared to provide strong typing in JavaScript alike in Java or C# programming languages. As a result, interfaces and models are implementable in Angular, that significantly simplifies work with objects and responses from remote servers. In Figure 19 example of such models is provided. For the work in the frontend itself, UI models of Movie and Genre were created. However, the backend of the application was returning respective objects with localization notation, which was also crucial in the frontend part of the project, therefore API models were included and used.

src > app > models > TS api-models.ts > ...

```ts
import { Movie, Genre } from './ui-models';

export interface MovieDTO {
    EN: Movie;
    FI: Movie;
}

export interface MoviesDTO {
    EN: Movie[];
    FI: Movie[];
}

export interface GenreDTO {
    EN: Genre;
    FI: Genre;
}

export interface GenresDTO {
    EN: Genre[];
    FI: Genre[];
}

```

src > app > models > TS ui-models.ts > ...

```ts
export interface Movie {
    id: number;
    title: string;
    year: number;
    poster: string;
    genres: string[];
    description: string;
}

export interface Genre {
    id: number;
    genreName: string;
    poster: string;
    movies: Movie[];
}

```

Figure 19. Model implementation through custom types - UI and API models

Here written explanation about Components, Modules, Services and navigation of Angular framework had a very basic form, yet provided concisely. The actual implementation

would have required too long and unnecessary explanation. For instance, Route objects can implement guarding, to prevent access and exit of a specific view, unless a specific condition is met. Such a feature can become useful for implementation of security measure or prevention of submission of potentially harmful data from a form that was inappropriately filled. However, such deep implementations of Angular framework were not required during the frontend part development.

Description of the Frontend application

The main goal of the frontend application, development of which was initialized before the development of backend, was to provide access to movie trailers for users from both mobile devices and personal computers. The interface had to be also user-friendly and support localization in both English and Finnish languages. The frontend development was managed using Git technology, and overall work done can be observed in figures below.

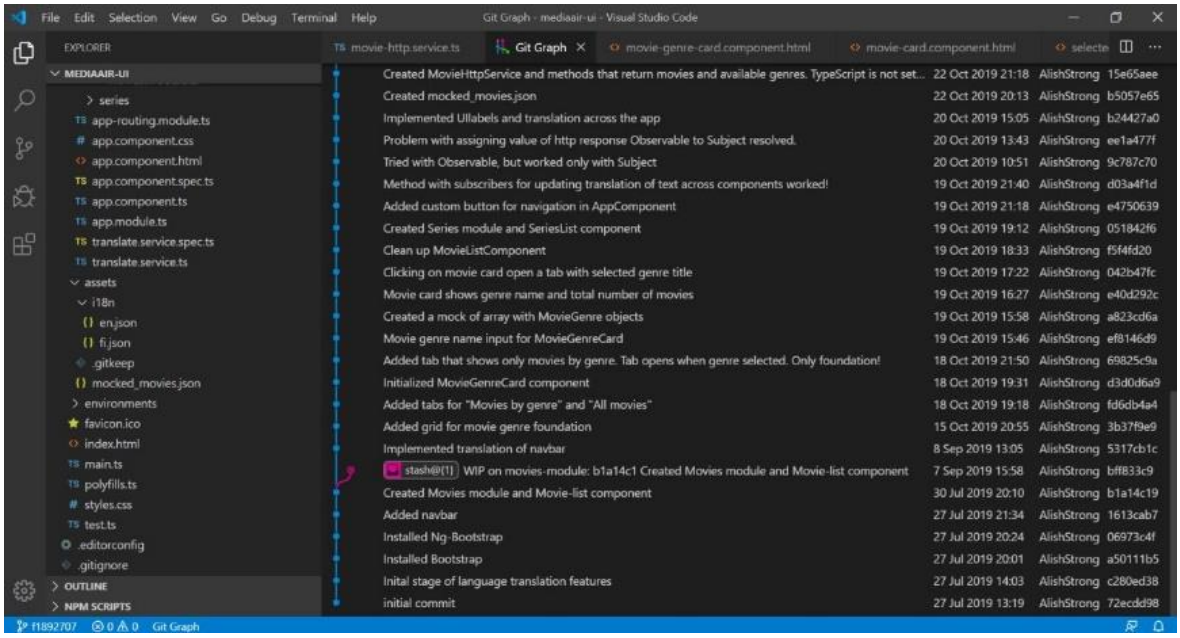


Figure 20. Git repository tree of the frontend application, part 1

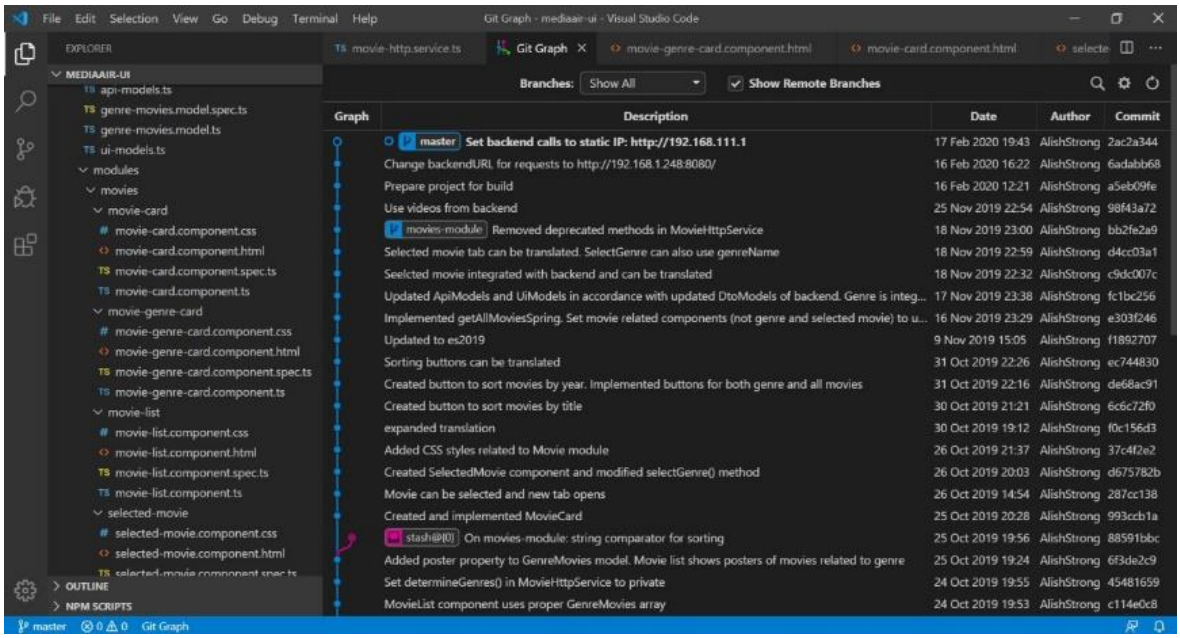


Figure 21. Git repository tree of the frontend application, part 2

As it is observed from Figures 19 and 20, work on the software started back in July 2019. General frontend body and functionality were achieved by the end of October 2019. In November 2019, work on the backend part of the web application commenced, and due to introduction of backend endpoints, modification of the frontend was also implemented. In total, the whole full-stack work was finished by the end of November 2019. However, the following months were dedicated to the preparation of the hardware and system infrastructure for the deployment. That is why when that work was done in February, additional changes had to be applied to the frontend and backend of the wIFE software.

To simplify the process of creation of an attractive user interface, Bootstrap library was chosen and installed using the npm; it was the third commit done and can be seen in Figure 20 with the hash of a50111b5. The decision for npm installation of Boostrap was linked to the Internet connection requirement of the traditional introduction of the library into a project. Since the wIFE had to work presumably in an offline mode in skies, library files had to be available locally. That is where the node package manager comes useful. It was later revealed, however, that a special adoption of Bootstrap for Angular existed. Namely, Ng-Bootstrap. It was also installed, nonetheless, the Angular adopted version of bootstrap was lacking proper documentation and therefore it was not widely used in the project. For example, the navigation bar on top of the application was created using the original Bootstrap library instead of the Angular adoption. Mentioning the navigation bar it should be added that although the stress was made to show only movie trailers, other tabs were also provided: series, music, and translation buttons.

Except for the movies and translation buttons, other sections in the bar were not fully implemented. Series section was done as a simple module with one component template. It was created to test the navigation in the application. However, since the movie series section would have had a similar design, its development was put aside. Music and Settings sections were also not developed, and instead were present in the navigation bar for esthetics purposes and additional tests of the navigation within the Angular application: upon pressing on them, internal routing mechanism had to redirect the user to Movies section. Since the navigation bar was created using Bootstrap library, it was completely responsive and transformed from a bar with navigation tabs and translation buttons on wide-screens of laptops and tabs (Figure 22) into an accordion menu on small-screen devices (Figure 23).

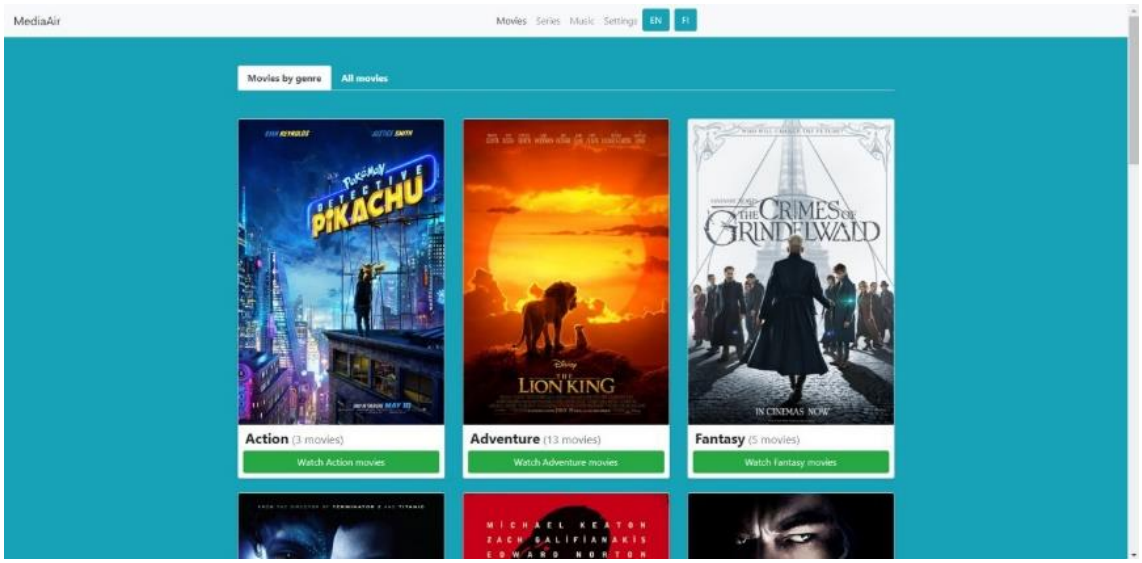


Figure 22. The user interface on a widescreen of a laptop

Figure 23. The user interface on a small-screen device - emulated iPhone X in developer tools of Google Chrome browser

The main developed section of the user interface was the Movies section. To implement that section of UI, a separate MoviesModule was created (Figure 24). The module declared four view components: MovieListComponent, MovieCardComponent, MovieGenreCardComponent and SelectedMovieComponent. If the navigation bar used the root RouterModule to navigate among menu tabs, the movie module implemented a different technique. Instead router navigation, component views were changed through tabulation.

The main parent component view in the module was MovieListComponent. The component implemented tab set and tab elements from Ng-Bootstrap library. The default tab view had an id of “genres” and consisted of MovieGenreCardComponent grid, that grouped movies by specific industry genre (Figure 22). Users could click on a card of a specific genre of interest and navigate to a genre tab, that included a grid view of movies that belonged to that genre. The grid in that tab consisted of MovieCardComponent elements. Since navigation was implemented through tab set and not routing, the address in the browser stood the same. Implementation of Angular Bootstrap ensured that unopened views and programming logic behind them would not be executed upon the access to MovieListComponent, therefore network-wise the application stood light-weight and could ensure smooth operation.

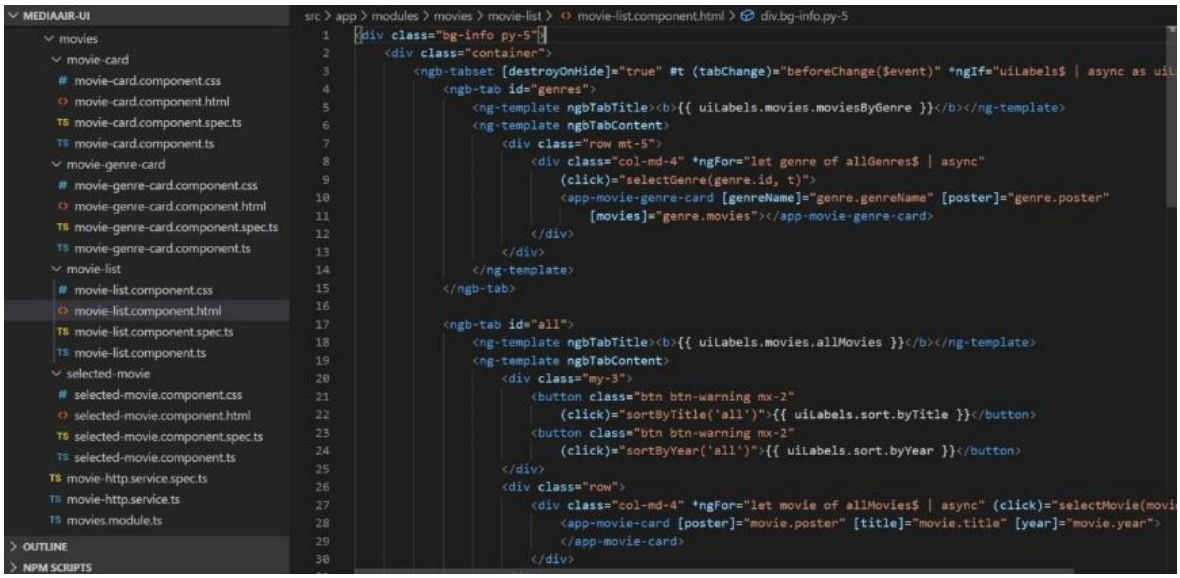


Figure 24. Movies' module and MovieList component

The MovieListComponent had another tab accessible for direct selection to users. The tab had an id of “all” and provided a grid of MovieCardComponent tiles. In that tab, movies were not grouped by their respective genre, instead, the user had a choice to view all available movie entertainment list of the wIFE. In order to simplify the browsing of the movie list, sorting by year and movie title was added to the tab. Two buttons were added to provide such functionality.

Tiles in “all movies” and “genre-specific movies” tab were represented by MovieCardComponent elements. The element showed a poster, title and year of release of a movie. Upon clicking on the element, a separate component view opened – SelectedMovieComponent. This component looked like a separate movie page. It included a description of the selected movie, genres to which it belonged and a video player element that allowed to watch the movie’s trailer. The genres were designed as clickable buttons; when a user clicked on one of them, a tab with a grid of all movies belonging to that genre would open, closing the movie’s tab at the same time. An important note, however, is that before the endpoints for movie content and data transfer were prepared on backed, a JSON string with mocked movies was used. Images for posters were obtained online. While the trailers were embedded from YouTube; that is illustrated in Figure 25.

Figure 25. SelectedMovieComponent with mocked movie data, online poster and embedded YouTube video

The mocked and later backend-fed data were obtained and provided by MovieHttpService. The service was located in the movies directory, along with the MoviesModule and its components. However, following the latest development recommendations, the service was not provided in MoviesModule or even the root AppModule. Instead, its provision was specified directly in the directive of @Injectable decorator, using providedIn property (Figure 17). In its final form, the service included methods to obtain movie or genre-specific data from the backend part of the application and supply that data to respective components of MoviesModule. It also implemented a method for sorting movies and a private comparator (Figure 26).

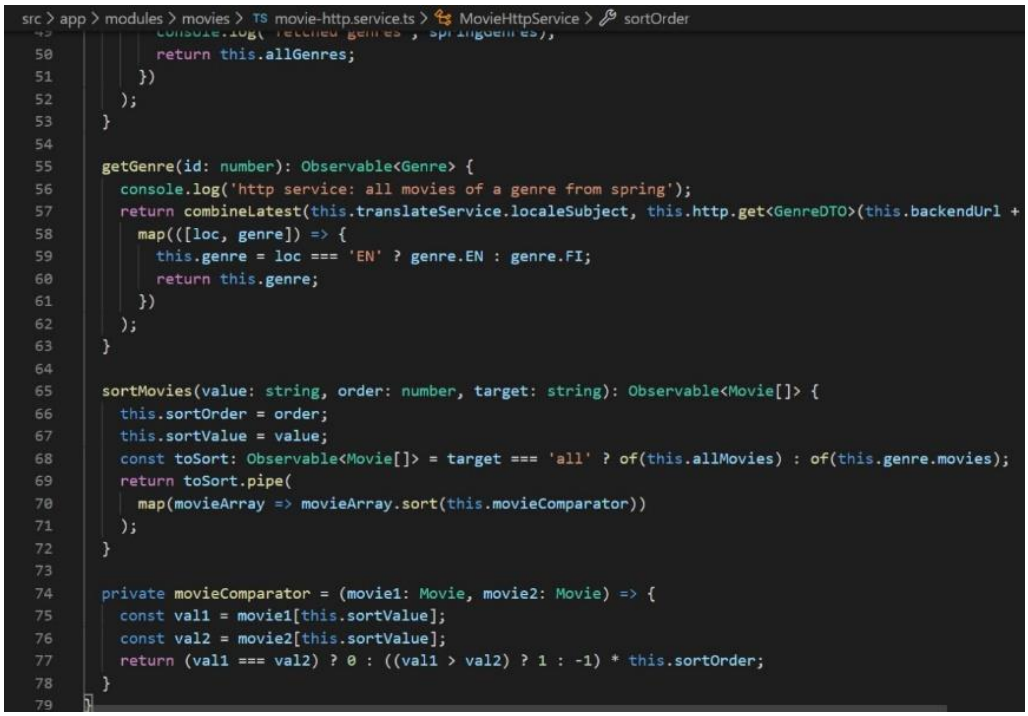


Figure 26. Sorting and Comparator implementation

A comprehensive work was done to implement localization with instant translation and preservation of the view location. For that purpose, a combined work was done on both backend and frontend, though the major implementation belonged to the frontend naturally. In the frontend project, two JSON assets were created. Both assets were JSON string objects with properties dedicated to the title, navigation bar, movies section and sorting buttons; each asset had an identical set of properties, yet different values, corresponding to the English and Finnish translation (Figure 27).

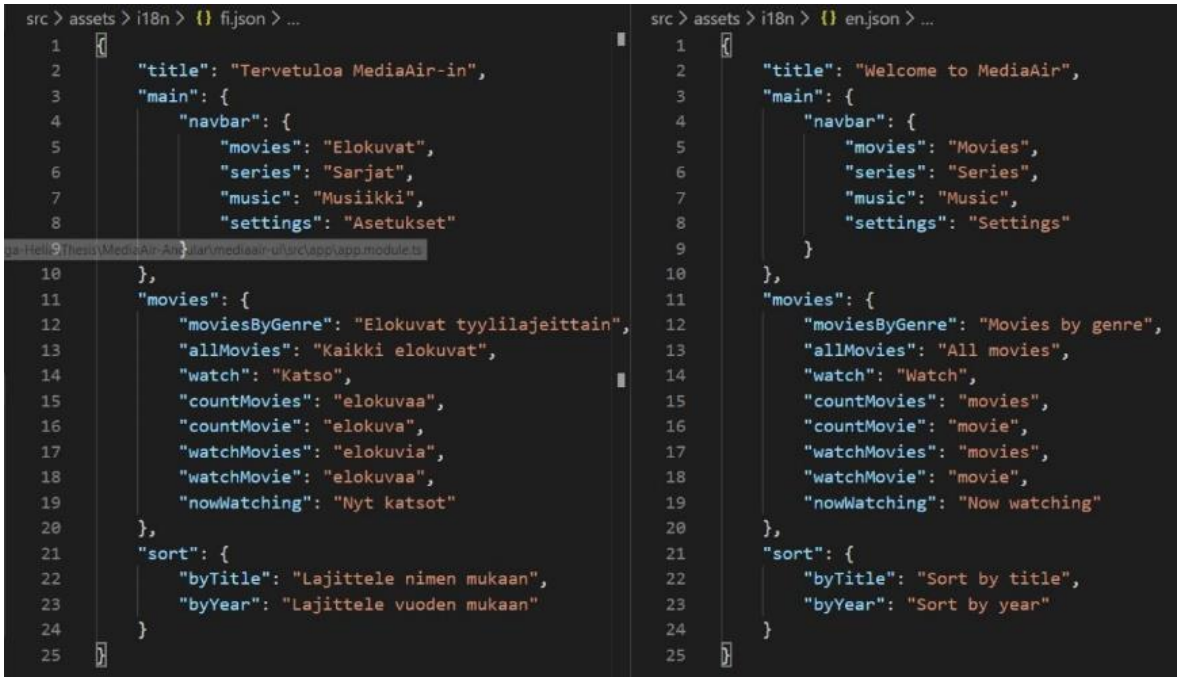


Figure 27. Localization assets

Localization assets were needed for translation of static textual elements of the user interface: title of the application, text in the navigation bar and action buttons. To apply either Finnish or English text into UI, TranslationService was used. The service can view observed entirely in Figure 28, and it used a simple yet interesting approach for instant localization of both static and dynamic data.

Figure 28. The translation service

The service had two public and one private properties. The private property named “locale” and a public property named “uiLabels” played the main role in fetching of asset data of right localization. As it was stated beforehand, the translation buttons were located in the navigation bar of the application, which in turn belonged to the AppComponent. The component had a translation method, which in fact was calling for “translate()” method of the TranslationService with the proper value for “locale”. The triggered, TranslationService would set the private property to the value passed and execute a request for static data fetch from the project’s asset. The value returned would then be assigned to the public “uiLabels” property. The public property had a type of a BehaviorSubject from RxJS library. RxJS library provides functional and rapid development capabilities to JavaScript. A crucial feature of BehaviorSubject, that was discovered during attempts of localization implementation, was immediate value firing to all elements that subscribed to it. Therefore, when a value change was triggered in AppComponent, resulted effect of localization change was available in all other components that subscribed to the same BehaviourSubject property of the TranslationService – MovieListComponent, SelectedMovieComponent. Again, “uiLabels” and “locale” properties impacted only the static content change. Before the creation of endpoints on the backend, localization buttons did not affect the mocked movie data, because the data was mocked in English

only. This can be observed in Figure 29, where localization was changed to Finnish, and except for the static elements of UI text, the movies-related text is provided in English.

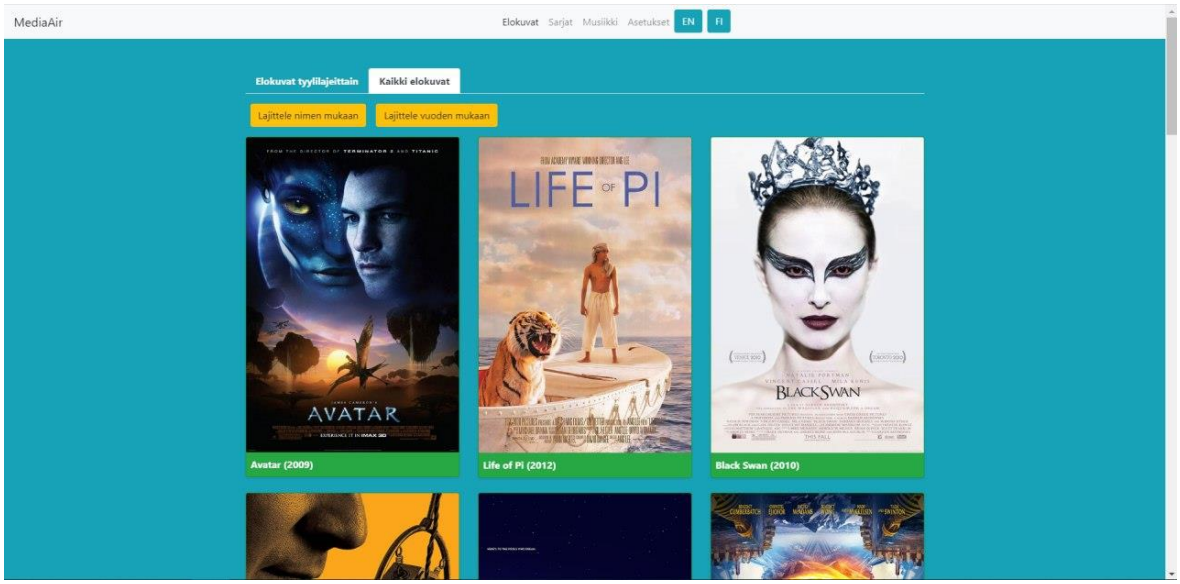


Figure 29. Localization under unfinished backend and mocked data: the static text is translated, while the dynamic text of movie titles is not

For the localization implementation of dynamic movie data both frontend and backend had to work in pair. The TranslationService had another public property – “localeSubject”, which was also a BehaviourSubject type. The “localeSubject” was mutated in the “translate()” method of the service as well, and the property was observed in the known MovieHttpService. Methods of the MovieHttpService implemented “combineLatest()” method from RxJS to monitor the state of the “localeSubject” and obtain data from backend request. RxJS “map()” operator allowed to react to the changeable state of “localeSubject” and read correct localization data from backends response. Backend’s endpoints were planned to return data grouped into localizations of English and Finnish language; when a request is made, for a specific movie, for example, an object string would be returned with data in both Finnish and English. However, depending on the “localeSubject” frontend was determining language data had to be set. Since the type of the property was also a BehaviorSubject, its value change was caught instantly by the MovieHttpService and therefore localization shift appeared instantly in the application without any need to refresh or change current page on the application.

These were the main aspects of the Angular application, developed as user interface and representing the frontend part of the wireless IFE project. As it was stated, the major development in frontend was achieved by the end of October 2019. Afterwards, work on the backend part of the project started to provide the endpoints for data provision, that

besides the content, should have supported localization. When the backend part was finished, additional changes were applied to frontend also for integration of the software as a whole and fitting into the deployment environment on CentOS6.

4.1.2 The building of Java application

Work on the backend of the web application began in November 2019. Targets of the backend application were the creation of REST endpoint for the frontend and ability to access multimedia files and their respective metadata to be fed to the frontend via the mentioned endpoints. It was decided to use the Java programming language for the development of the backend under Spring Boot framework. Development of the backend took around one month. After the main work on it was finished, a parallel work commenced on integrating the frontend to the endpoints and building two projects into one executable file. Initially, the unification of backend and frontend was planned through methods described by Majd Asab and Swathi Prasad. However, it was revealed that the methods were not applicable to relatively large projects and were prone to build failures. Instead, another building technique was developed, that allowed obtaining the desired executable file for deployment. The complete project structure and commits of the project's git repository can be viewed in Figure 30.

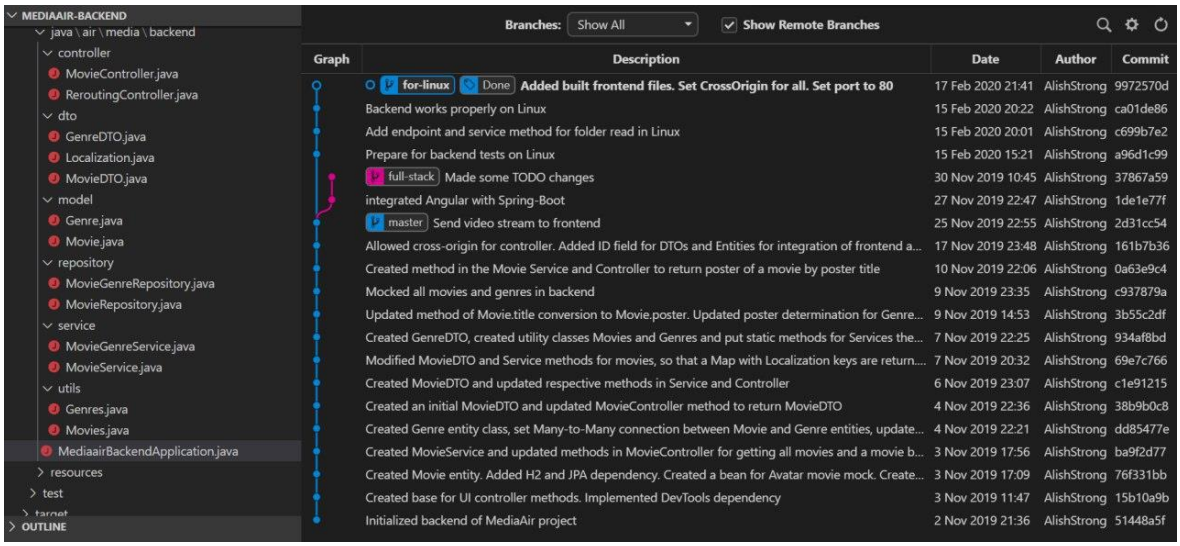


Figure 30. Complete backend project structure and its repository git tree

Developing projects in Spring Boot

Spring Boot is a Java framework that itself is based on the original Spring framework but with abstaining from prevailing extensible markup language (XML) configuration. Spring framework appeared to simplify the process of enterprise programming in Java. Spring Boot, on the other hand, appeared to simplify the use of Spring itself. Advantages of

Spring Boot is full programmability in pure Java object-oriented programming, though it is still possible to use XML configuration files, pre-configured application server – Apache Tomcat, which simplifies the deployment, pre-configuration of major Spring libraries, and excellent documentation and community support (VMWare Inc., 2015). Important to note, however, that Spring framework is not the only Java framework; there are other frameworks, like Apache Spark, Google web toolkit, Java Server Faces from Oracle and even an original Finnish framework – Vaadin (Vaishnavi, 2020).

When Spring projects are mentioned, it is important to raise the subject of Maven. Maven is a build and dependency automation tool, somewhat similar to npm (The Apache Software Foundation, 2020). Not all Java projects, that use Maven are Spring projects. But Maven allows using Spring and Spring Boot dependencies, which then, along with other important aspects in code, turn the project into a Spring or Spring Boot project. Also, important to note that Spring Boot should not necessarily use Maven, it is possible to use the Gradle tool instead.

Spring framework allows creating web applications using the MVC framework – Model, View, Controller (Figure 31). It also includes dependency injection and Service classes. Because of this, it is possible to create full-stack projects purely with Spring or Spring Boot frameworks. However, since Angular was used for the user interface development, the view part of Spring Boot regarding the thesis project was not included in the backend[2]. Models represent classes the state of which could change. Views are responsible for feeding user with what has to be shown. Controllers make sure that Views get the right data, and that Models are mutated with respect to actions in Views. Service classes and dependency injection appeared due to the need of cross-controller service provision. It can be observed that paradigms of development in Angular are very similar to the MVC framework under Spring's scope. That is why Angular and Spring Boot projects can often be found in public repositories (although as separate backend and frontend projects, deployed separately).

[2] It actually was, but in a special technique discovered to integrate Angular with Spring Boot into one build.

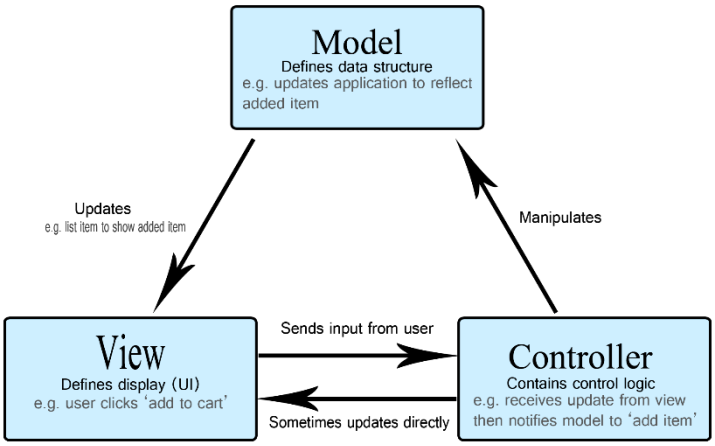


Figure 31. Model-Movie-Controller framework (Mozilla, 2019)

Backend project overview

The backend project had to provide REST endpoints for the frontend part of the wIFE software. Since it was decided to exclude other types of media, except for movies, that could be categorized by genre, the model directory of backend included two model classes only: Genre and Movies (Figure 32). Although no database setup on the deployment server was planned, the models still extended Entity framework of Java Persistent API (JPA). The reason was in the internal use of the H2 database in the backend of the project, and the organizational simplification the JPA provided. For instance, a movie could belong to different genres; genres could include different movies – such relation is managed in JPA through @ManyToMany annotation, observed in Figure 32 in Genre and Movie classes. Such annotation simplifies the process of retrieving all movies of a genre or determining all genres to which a movie belongs.

src > main > java > air > media > backend > model > Genre.java > {} air.media.backend.

```
@Entity
public class Genre {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String nameEng;
    private String nameFin;

    @ManyToMany(mappedBy = "genres", fetch = FetchType.LAZY)
    @JsonIgnore
    private Set<Movie> movies = new HashSet<>();

    protected Genre() {}

    public Genre(String nameEng, String nameFin) {
        this.nameEng = nameEng;
        this.nameFin = nameFin;
    }

    public Long getId() {
        return id;
    }

    public String getNameEng() {
        return nameEng;
    }

    public String getNameFin() {
        return nameFin;
```

src > main > java > air > media > backend > model > Movie.java > {} air.media.backend.model

```
@Entity
public class Movie {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private Integer year;
    private String poster;

    private String titleEng;
    private String titleFin;
    private String descriptionEng;
    private String descriptionFin;

    @ManyToMany(fetch = FetchType.LAZY)
    @JoinTable(
            name = "Movie_Genre",
            joinColumns = @JoinColumn(name = "movie_id"),
            inverseJoinColumns = @JoinColumn(name = "genre_id"))
    private Set<Genre> genres = new HashSet<>();

    protected Movie() {}

    public Movie(Integer year, String titleEng, String titleFin, String
        this.year = year;
        this.poster = titleToPoster(titleEng);
        this.titleEng = titleEng;
        this.titleFin = titleFin;
        this.descriptionEng = descriptionEng;
```

Figure 32. Genre and Movie model classes

Entity models, especially with complex relations, should be handled cautiously, however. If a controller responded directly with an instance of a genre, an endless loop would be observed: both Genre and Movie entity classes are linked through many-to-many relation; a Genre object would have a set of Movie objects, each of which would have a set of Genre object and so on. To prevent such a scenario, a common practice of Data Transfer Objects – DTO, was applied. DTO is a separate class that typically wraps the input object and disconnects it from redundant properties and thus becomes more applicable for the HTTP response. For the backend, GenreDTO and MovieDTO classes were created (Figure 33). While the GenreDTO still included a set of MovieDTO objects, MovieDTO on other hand had a substitution of object genre set to a list of genre names as string values. That meant, that when a genre was requested, the DTO response would include a set of movies inside of the genre, but movies themselves would refer to genre names only, thus the infinity caused by many-to-many relation would be prevented.

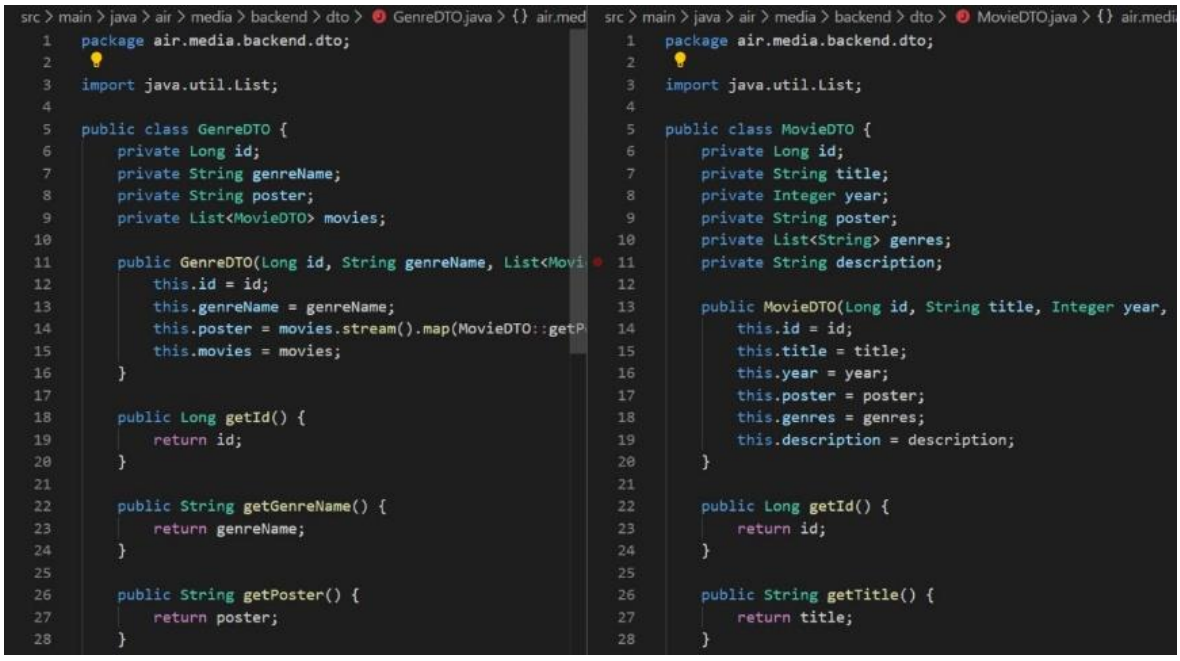


Figure 33. GenreDTO and MovieDTO

The REST endpoints needed for the frontend were specified and managed in the MovieController of the Spring Boot application (Figure 34). However, the controller was not responsible for fetching the genre or movie data, especially in DTO format; the only direct responsibility for the controller was obtainment, read and feed of video file of a requested movie in “getMovieVideo()” method. The controller, instead, was serviced by two respective service classes: MovieService and GenreService (Figure 35). These services were obtaining Movie and Genre entity objects with help of respective repository classes and then using specially designed static classes were transforming entities into

DTOs. The utility classes of Genres and Movies (Figure 36) were transforming not simply the entity objects into DTOs, they were preparing DTOs in two localizations (English and Finnish) and wrapping them into a map, that was then returned to services, from services to the controller, and from the controller to the frontend. Localization enum class was intentionally created to assist the static classes.

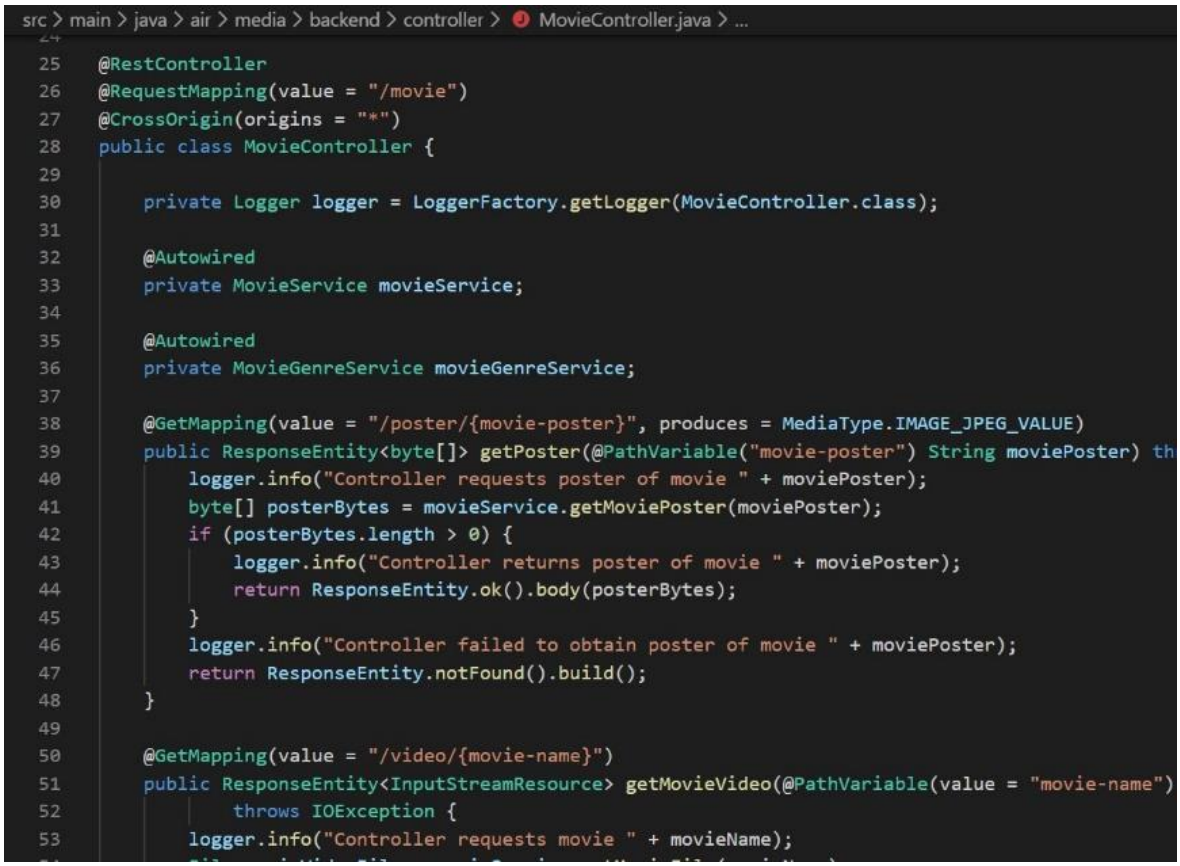


Figure 34. Movie controller

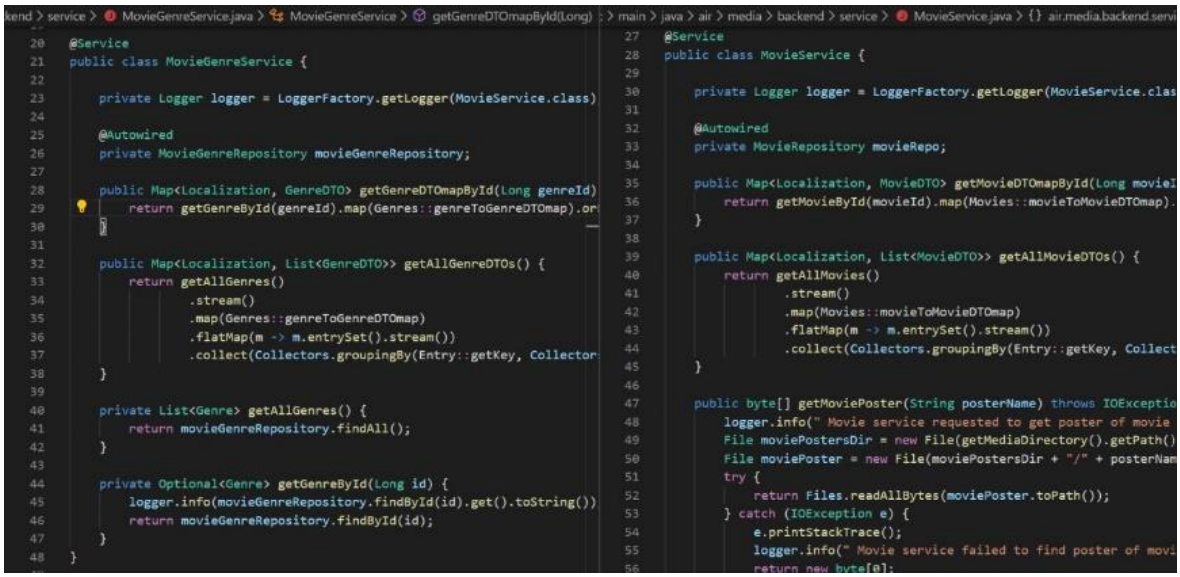


Figure 35. Service classes

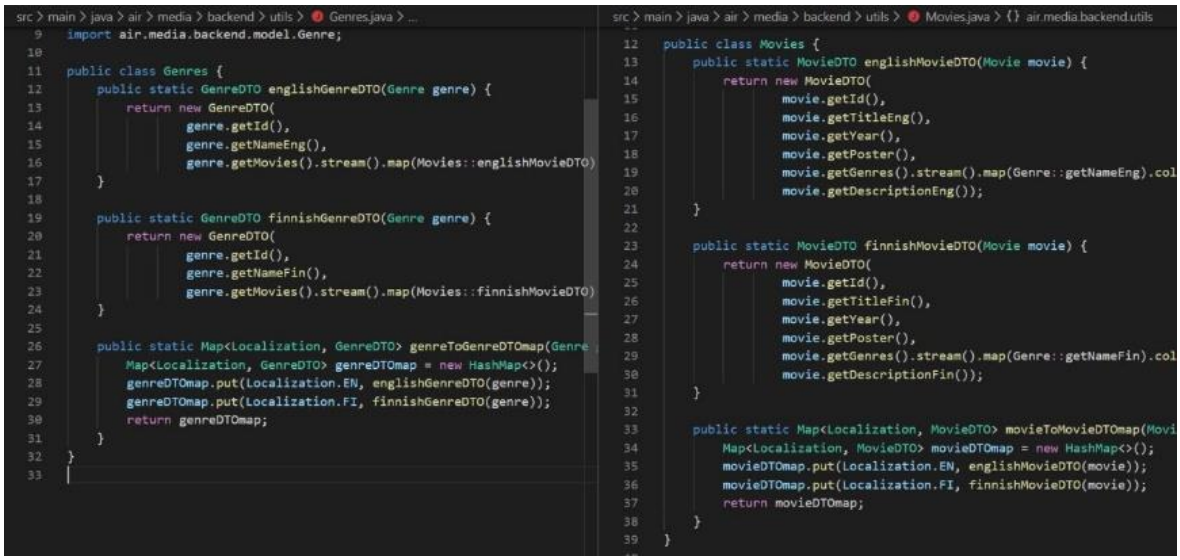


Figure 36. Utility classes

Development of the backend required less time than the development of the frontend part. By the end of November 2019, it was ready for integration with the front end and overall project build. However, it is crucial to highlight that the development ease was related to simplifications applied to the project's functionality on both frontend and backend side. Even with simplified functionality set, without full-stack integration, development took over four months of work. The project would have taken much more time, if at least instead of mocking data on the backend side through entity beans and H2 database, actual database setup and configuration was included. Nonetheless, with major completion of backend and frontend, integration of two parts could be started.

4.1.3 Integration of Angular with Java

Having backend and frontend ready, it was possible to commence the build of the project into one executable file. The file had to be either Java executable: either of .jar or .war extension. Maven simplifies the process of the project builds and by default generates .jar files. However, a simple build of java projects would result in backend executable files, with frontend missing. Angular, using Angular CLI, creates its own built files meant for deployment. Therefore, there was a problem faced in the integration of two projects into one. Traditionally, two builds would be deployed separately – modular approach in software development, but such scenario was not favourable, because even though it was possible to deploy the builds on the same server, wIFE start-up would require two separate starts, which deemed to be inconvenient. Nonetheless, a method to achieve the desired one-build goal was found and applied.

The method used to integrate frontend with backend was discovered during the work and applied in the project. However, prior to its discovery attempts to use two other methods, found on the Internet, were made. The methods, however, resulted in slow builds and were prone even to errors at the building process. Both methods were using additional plugins in the project's Maven configuration, though the type and number of them were different.

The first method was described by Swathi Prasad (2018). First of all, Prasad followed the proper module architecture of Maven in her example project. There were a parent and two child projects (Figure 37). Specification of parent-child relation was properly set in pom.xml configuration files of the projects. Crucial to indicate that Prasad created three maven projects in her example. Although the system consisted of a parent, the parent worked as a joint, rather than a build unifier in Prasad's project. The main action was happening in two children projects: tutorial-web – frontend, and tutorial-server – backend (Figure 38). Again, all projects were Maven projects; frontend was an Angular, wrapped in Java-Maven.

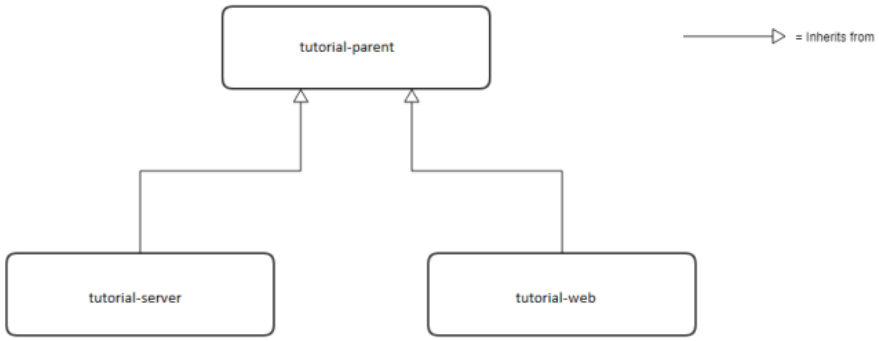


Figure 37. Structure of Prasad's project (Prasad, 2018)

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.swathisprasad.tutorial</groupId>
    <artifactId>tutorial-parent</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <packaging>pom</packaging>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.0.4.RELEASE</version>
        <relativePath />
    </parent>
    <modules>
        <module>tutorial-server</module>
        <module>tutorial-web</module>
    </modules>
</project>
```

Figure 38. pom.xml file of Prasad's parent project (Prasad, 2018)

The tutorial-web project had a web directory, where Prasad initialized an Angular project. The main step of her frontend configuration was done in pom.xml file of the Maven project (Figure 39). Prasad used frontend-maven-plugin created by Eirik Sletteberg – eirslett (GitHub, Inc., 2013). The plugin allowed to execute npm commands in Maven, which Prasad did. Instead of using Angular CLI commands, Prasad instructed Maven to use npm to install NodeJS and npm, install all dependencies mentioned in package.json of her Angular project, then build the project for the production environment.

```
<plugin>
    <groupId>com.github.eirslett</groupId>
    <artifactId>frontend-maven-plugin</artifactId>
    <version>1.3</version>
    <configuration>
        <nodeVersion>v8.11.3</nodeVersion>
        <npmVersion>6.3.0</npmVersion>
        <workingDirectory>src/main/web/</workingDirectory>
    </configuration>
    <executions>
        <execution>
            <id>install node and npm</id>
            <goals>
                <goal>install-node-and-npm</goal>
            </goals>
        </execution>
        <execution>
            <id>npm install</id>
            <goals>
                <goal>npm</goal>
            </goals>
        </execution>
        <execution>
            <id>npm run build</id>
            <goals>
                <goal>npm</goal>
            </goals>
            <configuration>
                <arguments>run build</arguments>
            </configuration>
        </execution>
        <execution>
            <id>prod</id>
            <goals>
                <goal>npm</goal>
            </goals>
            <configuration>
                <arguments>run-script build</arguments>
            </configuration>
            <phase>generate-resources</phase>
        </execution>
    </executions>
</plugin>
```

Figure 39. Plugin configuration in the tutorial-web project (Prasad, 2018)

The built files of tutorial-web were destined for the tutorial-server – backend part of Prasad's project. The backend, through the parent project and goals of maven-resources-plugin, was able to copy built files from the tutorial-web project and put them into the tutorial-server's resources folder of the built (Figure 40). Hence, when the build execution

command was given to Maven, it would first perform actions stated in tutorial-web project, then it would copy the files from frontend and include them in the build of the backend project, which would result in a single executable file (Prasad intentionally specified .war extension in her example).

```
<plugin>
    <artifactId>maven-resources-plugin</artifactId>
    <executions>
        <execution>
            <id>copy-resources</id>
            <phase>validate</phase>
            <goals>
                <goal>copy-resources</goal>
            </goals>
            <configuration>
                <outputDirectory>${project.build.directory}/classes/resources/</outputDirectory>
                <resources>
                    <resource>
                        <directory>${project.parent.basedir}/tutorial-web/src/main/web/dist/np-app/</directory>
                    </resource>
                </resources>
            </configuration>
        </execution>
    </executions>
</plugin>
```

Figure 40. Tutorial-server project's pom.xml (Prasad, 2018)

Method of Prasad was tested on new test projects. The method worked, however, disadvantages were easily observed. The first point was not a disadvantage, but a project structure improvement suggestion. The parent project in Prasad's architecture did not play any significant role other than chaining frontend and backend modules. Therefore, it was possible to use only two projects, where the backend was a parent, and frontend was a child. The main disadvantage was related to time consumption, especially in the tutorial-web project. Arguably, Prasad's solution was a suitable DevOps automation technique, where backend and frontend had different environments, yet the deployment had one major pipeline. In such a scenario it was wise to redownload and reinstall all packages for frontend, so that backend build included the latest changes of the frontend. However, with further development in the frontend, execution of those scripts would consume more and more time. As a result, when applied on the thesis's project, build required a considerable amount of time. Another issue was, that whenever changes were done in the frontend, to tune it for proper connection with backend, new Maven rebuild was required. Therefore, Prasad's solution was not suitable for the application in the wIFE project.

Another solution found on the web was described by Majd Asab (2019). Asab's solution was less complicated in terms of structure and involved only one Spring Boot project, inside of which an Angular project was nested. In the resources directory of the Spring Boot project, Asab created a "frontend" folder, in which he initialized an Angular

application using Angular CLI. If in Prasad's solution all configurations were done on the Maven side (through pom.xml files), Asad also indicated the need for configuration of the Angular project. Namely, in the angular.json file, it was necessary to set the output path of the project's build to a public directory inside the resource folder of the Spring Boot's project. Afterwards, configurations were applied to the pom.xml of the backend side. Asab used exec-maven-plugin from MojoHaus community. The plugin allowed to specify additional execution of non-Maven scope in different phases; instead of specifying those execution commands in command line together Maven's main build commands, pom configuration could be made, with phase level specification, and those executions would be done prior the main build (MojoHaus, 2017). Asab specified execution of Angular CLI build command on validation phase (Figure 41). The result was working in the following way:

- First, exec-maven-plugin would build the Angular project;
- Specification inside angular.json would put the built files into resources/public folder of the Spring Boot project;
- Finally, Maven would build the Spring Boot project itself, that by the time of build would already have Angular files. As a result, an executable file will be obtained with necessary frontend components included;

```
<build>
    <plugins>
        <plugin>
            <groupId>org.codehaus.mojo</groupId>
            <artifactId>exec-maven-plugin</artifactId>
            <version>1.6.0</version>
            <executions>
                <execution>
                    <phase>validate</phase>
                    <goals>
                        <goal>exec</goal>
                    </goals>
                </execution>
            </executions>
            <configuration>
                <executable>ng</executable>
                <workingDirectory>src/main/resources/frontend/angular-app</workingDirectory>
                <arguments>
                    <argument>build</argument>
                </arguments>
            </configuration>
        </plugin>
        <plugin>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-maven-plugin</artifactId>
        </plugin>
    </plugins>
</build>
```

Figure 41. Configuration of pom.xml in Asab's solution (Asab, 2019)

Asab's method was first tested on newly created Spring Boot and Angular projects, and the method indeed work. However, one of its main issues emerged with the increasing complexity of the Angular application. Builds were often failing due to the use of the

unstable plugin, that was last updated back in 2017. Another issue of Asab's method was related to the decision of nesting the Angular application in the Spring Boot application. Angular projects require a vast number of packages by default. Those packages are downloaded and stored in the node_modules directory of the project. Due to the sheer number and volume of packages needed for Angular applications, the size of the directory can be over 250 megabytes (Figure 42). While the execution of a command to build the frontend was not a problem, the building of the Spring Boot project faced an issue, due to the mere enormous project size, impacted by the node_modules directory of the nested Angular application. In the best outcome, the project took over fifteen minutes to build, however, most of the time building was failing when Asab's method was applied to the actual project of the thesis work.

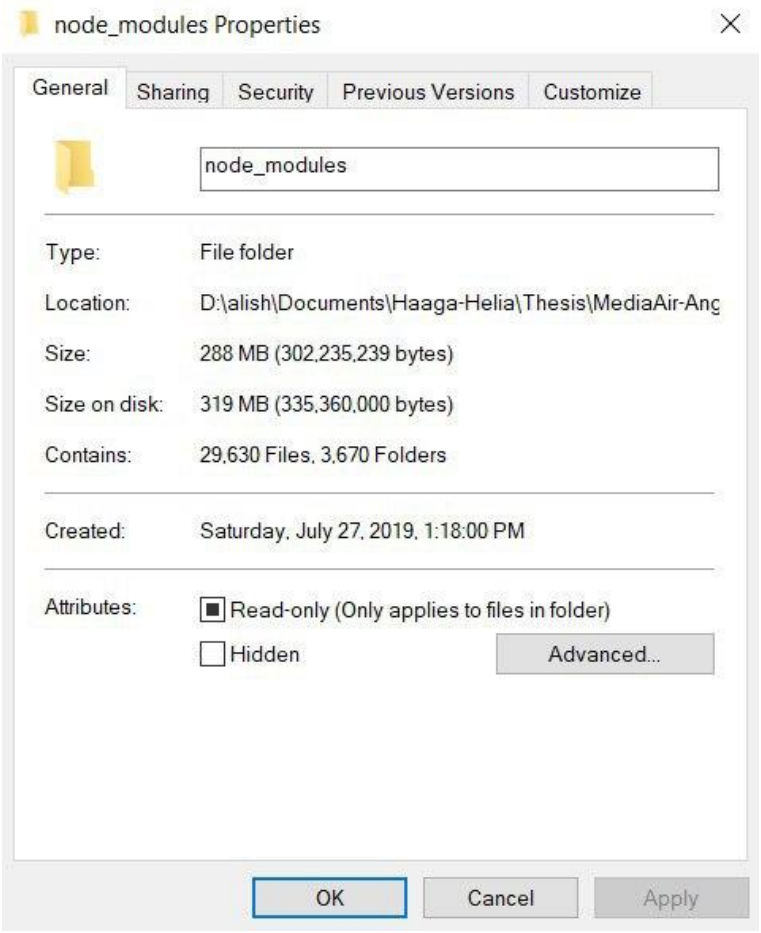


Figure 42. Size of node_modules directory in the frontend of the thesis project

Although the methods of Asab and Prasad revealed to be inapplicable for the thesis project, they pointed to hits of how to integrate frontend with backend and build two sides of the full-stack project into one file. The solution was based on the way Angular and Spring Boot applications work.

Firstly, additional information about the Angular framework should be given. The Angular framework allows to created single-page applications, known under abbreviation SPA. Such applications consist only of one main HTML document, that is typically named index.html; the main logic and functionality is performed by JavaScript code for Angular is a JavaScript framework[3] (Goel, 2019). That is, when an Angular project is built, in a dedicated output directory, an index file and required JavaScript files would be generated. Opening of the index file would trigger the start of the JavaScript code which would initialize the application and show it respective user interface in a browser.

Secondly, Spring Boot framework was created to simplify the process of application development in Spring framework. Spring Boot has a lot of configurations prepared in advance, many of which are specified in “WebMvcAutoConfiguration” (Syer, 2014). One of the automations is projected on the default mapping of domain requests to static resources in the classpath of a Spring Boot application, which is by default is a resource folder in the standard Maven project structure with additional possibilities of storage in public and static directories in that same folder (Pivotal, 2020). That means, that if in either resources-folder or resource/static-, resource/public-folders an index.html is present, Spring Boot would automatically map all empty, non-parameterized domain requests to the output of that index file (Java Developer Zone, 2017).

That is why Prasad copied build files of the Angular project to the resource folder of her tutorial-server project, while Asab set the build output directory in his Angular application to the resource/public folder. Therefore, instead of applying a significant endeavour with different Maven plugins, the simplest solution was to build an Angular application first. Then copy the build files of the frontend project to resources folder in Spring Boot; in the thesis project, the files were copied to resources/public-folder (Figure 43). The final execution of Maven build command would result in a java executable file with integrated frontend and backend.

[3] Another reminding note could made about the fact that Angular apps are nested trees of component templates.

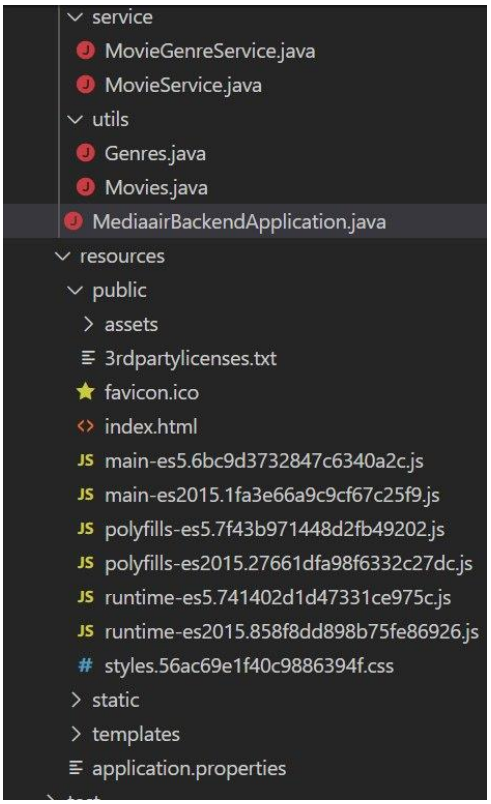


Figure 43. Built Angular files in resource/public folder of the backend project

Changes in frontend and backend

The method of backend and frontend integration was successfully applied to obtain the java executable file. However, both the frontend and backend parts required additional modifications. On the backend side, ReroutingController was added to handle navigation issues observed in the user interface after the build (Figure 44). Controller solution was provided by Eduardo Eljaiek (2017), who pointed for the need of backend-side forwarding as a solution for a similar problem observed.

src > main > java > air > media > backend > controller > ReroutingController.java > ReroutingController

```
1  package air.media.backend.controller;
2
3  import org.springframework.stereotype.Controller;
4  import org.springframework.web.bind.annotation.RequestMapping;
5
6  @Controller
7  public class ReroutingController {
8
9      /**
10      * Smoothly handles re-routing, which allows close integration of Angular
11      */
12     @RequestMapping(value = "/**/{[path:[^\\.]*}")
13     public String redirect() {
14         return "forward:/";
15     }
16 }
17
```

Figure 44. Rerouting controller that preserved navigation in Angular UI

On the frontend side, instead of mocked files, actual requests to backend had to be established, which was specified. However, it was revealed that additional configuration had to be applied for the MovieController in the backend. Integration of frontend with backend resulted in REST calls coming from the application and addressed to the application itself. Such behaviour is prevented by many systems by default, including the Spring framework. It is needed to prevent potentially harmful requests, that could overload the system. However, the method switching that safety switch was very simple: MovieController required a @CrossOrigin annotation with “origin” parameter set to any source using the asterisk sign according to wildcard notation (Figure 45).

```
24
25  @RestController
26  @RequestMapping(value = "/movie")
27  @CrossOrigin(origins = "*")
28  public class MovieController {
29
30      private Logger logger = LoggerFactory.getLogger(MovieController.class);
31
32      @Autowired
33      private MovieService movieService;
34
35      @Autowired
36      private MovieGenreService movieGenreService;
```

Figure 45. CrossOrigin annotaton in MovieController class

4.2 Integration of the software with the hardware

After the software part of the thesis project was finished, work on hardware and system commenced. Software required a deployment and run environment, which had to be provided by the main computer. The main computer had to be linked to a wi-fi router. The router would create a private network, to which any user would be able to connect. While the system would not have Internet accessibility, it would allow to open an entertainment portal in a web browser and watch media content stored on the server and provisioned by the running web application.

Main hardware components were a personal laptop computer and a wi-fi router purchased for the project. Since the computer was running on Windows 10 operating system, and for the purposes of simulation of a microcomputer use, CentOS6 software was virtualized as the system's main server. The router device also had to be configured properly for the needs of the project. Finally, changes on backend and frontend part of the web application required adaption to settings of actual system configurations.

4.2.1 Server-side configuration

As it was mentioned, hardware and system set-up and integration work included preparation of a CentOS6 environment. CentOS6 is the sixth version of the Linux-based operating system, that was created and has been managed by The CentOS Project community. The software is open-source, is freely available for download and use and is positioned as an in-between solution of Fedora Linux and Red Hat Enterprise Linux, thus having many features of both public and enterprise solutions on the web (The CentOS Project, 2019). Reason for selection of CentOS and specifically of version six was based on previous experience of work with the software and therefore availability of basic knowledge of its proper use. Besides, later versions of the software had significant changes, explorations of which could consume a valuable amount of time and hinder project's progress.

To employ capabilities of CentOS6 for the project, a virtualization software was required. VirtualBox software from Oracle was selected and installed for that purpose. VirtualBox project is freely available for professional and personal use and is open-source software, even though it is backed and distributed by Oracle (Oracle, 2007). The software had been used before and its selection was justified by experience as well.

For the project, a minimal version of CentOS6 image was downloaded from repositories of The CentOS Project community and it was used to create a new instance of a virtual machine on VirtualBox software. The virtualized software had to directly use the wi-fi router of the wIFE system. It was also virtualized, and therefore required a connection to its terminal interface through another third-party software[4] – PuTTY, open-source software allows to establish a connection to remote machines and provides emulation of their terminals with pleasant text font (PuTTY project, 2018). Moreover, for the work of the whole wIFE, additional software had to be downloaded and install on the CentOS6 instance – Internet connection to the machine, for the project development purposes, was necessary.

Thus, during the installation process of the operating system, two network interfaces were specified. The first network interface – eth0, was used for the Internet and terminal emulator connections; it had to obtain an IP address dynamically at the on-boot process of the instance. The second network interface – eth1, was dedicated to the connection with

[4] Terminal window from VirtualBox was esthetically unpleasant.

the router and main operations related to providing entertainment access to connected users; it hat to be static and was assigned 192.168.111.1/24 (the later part of the address, after the dash, is a sub-mask indication). Other important actions to be done at the installation of the CentOS instance was to select correct time-zone – UTC+2 (Helsinki), interface language – English, keyboard pattern – Finnish, and domain – media.air.

After the installation was finished, an update of the system could be performed using “yum update -y” command. Moreover, different utilities had to be installed to provide core functionality for the server and simplify the process of work on the virtualized machine. Using a single command on the terminal “yum install mc ntsysv dhcp bind bind-utils -y” following utilities were installed:

- MidnightCommander – a visual file manager that provides a convenient interface for work with files, especially on terminal systems like CentOS (Trac, 2020);
- ntsysv – a utility that provides an interface to control services and their activation or deactivation on Linux-machines (Red Hat, Inc., 2006);
- DHCP - Dynamic Host Configuration Protocol service that would allow the virtual machine to be turned into a DHCP server and thus control IP-address assigning process of connected users (Kili, 2018);
- DNS – Domain Name System service to provide a translation of computer address to human-untestable addresses and vice-versa: bind and bind-utils provide utilities for DNS services (Both, 2017);

Another software that had to be installed on the server was Java itself because the developed web application was a java executable. Instead of using an IBM or Oracle version of Java, a community-supported OpenJDK was installed. The version selected was eight – one of the most stable versions of Java. In order to install the software following command was executed in the machine’s command line: "yum install java-1.8.0-openjdk" (Oracle, 2011).

For assurance of application of correct configurations at the installation stage of the operating system, the hosts file could be checked (Figure 46). The file is located in /etc directory of the system.

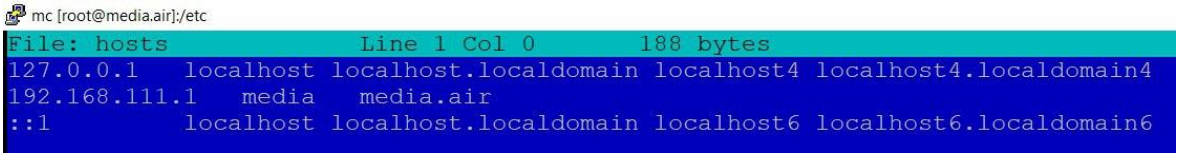


Figure 46. Configuration of the hosts file

The system, that imaged to operate in a cabin of an airplane, was not planned to provide Internet access onboard. Therefore, some security measures of the machine could be turned off. On many Linux products, CentOS up to version six in particular, security module logic is controlled by SELinux, the configuration file of which could be found in /etc/selinux directory. Selinux is an important module that protects machines from threats that have Internet-origin. However, it could cause additional inconveniences for the project's system and therefore was disabled in the installed instance of CentOS6 (Figure 47).

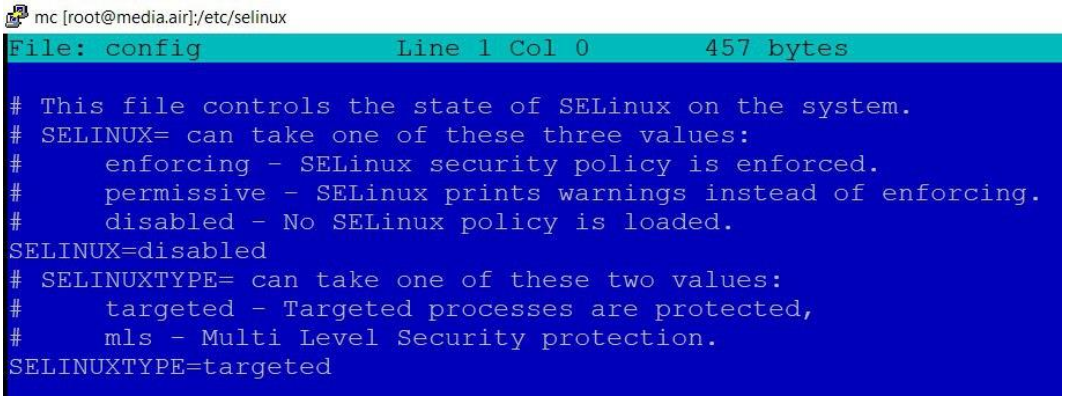


Figure 47. SELinux shutdown in the respective config file: SELINUX=disabled

The CentOS6 instance had to be the main server of the developed wireless IFE system. It had to be an environment for the web application, it had to store the content files and it had to service access to the system and its network, created, theoretically, by several wi-fi routers. That is why DHCP, DNS and Java software and utilities were installed. The first step in the server's configuration was set up of DHCP capabilities.

Setup of DHCP server capabilities on Linux machines was explained by Aaron Kili (2018). As it was mentioned and done, dhcp service itself was installed on the server. The next crucial step was the configuration of two main files: dhcpd and dhcpd.conf. File dhcpd is located in /etc/sysconfig directory and specifies the interface for requests management. Since eth1 was created for that purpose, it had to be added to DHCPDARGS value in the file (Figure 48). Such a setting would ensure IP address sharing on eth1 interface only.

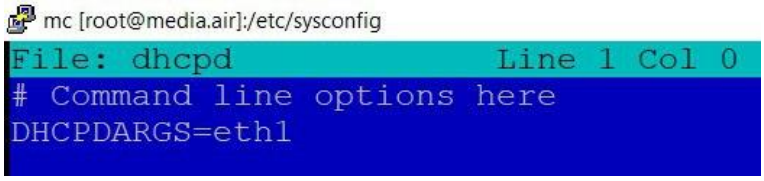


Figure 48. Setting DHCP service on eth1 network interface

Configuration of the DHCP server logic itself had to be done in dhcpd.conf file, which could be found in /etc directory (Figure 49). The main aspects in the file were an indication of the domain name, its server and sub-network. Domain was “media.air” and its server address was set to 192.168.111.1 – because it was planned that the server would perform DNS capabilities as well. The range of addresses to give to connected wi-fi routers was in between 192.168.111.2 and 192.168.111.254, so that up to 252 devices could connect to the network; GoGo Vision used nine wi-fi routers, while the actual implementation included only one router.

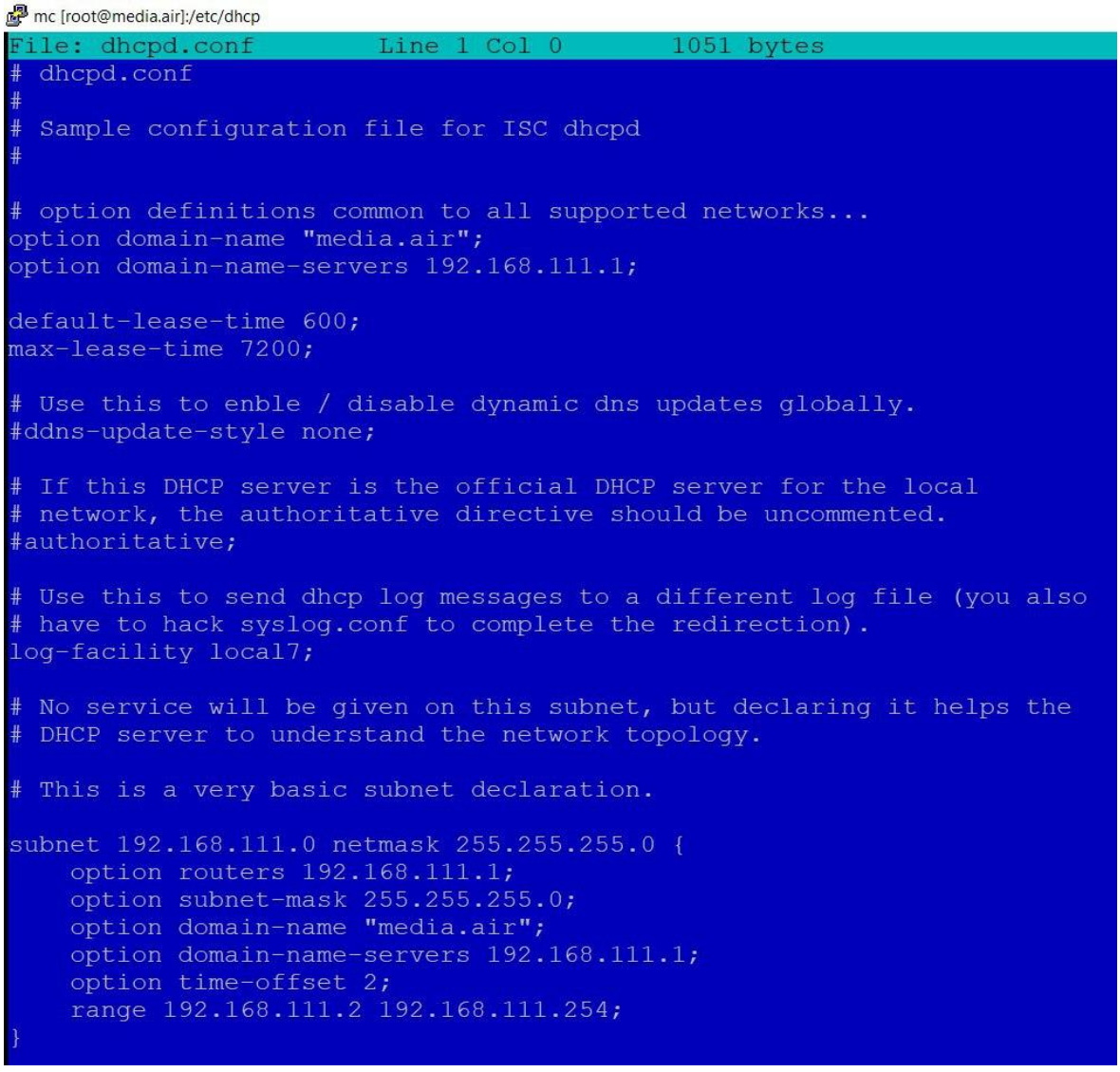


Figure 49. Configuration of DHCP server in dhcpd.conf file

The next step in server preparation was setting the Domain Name System – DNS. For that purpose, bind and bind-utils packages were installed. They were needed to provide NAMED service on the server. However, further work was needed. In DNS server setup, explanations by David Both (2017) were used, though, upon the understanding of the setup logic, a different approach was used. The first configuration file was named.conf in

/etc directory (Figure 50). It was important to specify the port and proper address to let other devices connected to use the service. The eth1 was responsible for the needs of wIFE operation, therefore IP was set to 192.168.111.1. The server also had to accept queries from connected devices. Since DHCP was configured on 192.168.111.0 network address, the value for the allow-query line was set to 192.168.111.0/24 (/24 was a respective subnet mask). Again, the actual Internet connection was not planned for the system, and therefore there was no need to specify external forwarders, like in Both's tutorial, thus that line was completely removed from the configuration. IPv6 support was also not needed, and its line was removed as well.

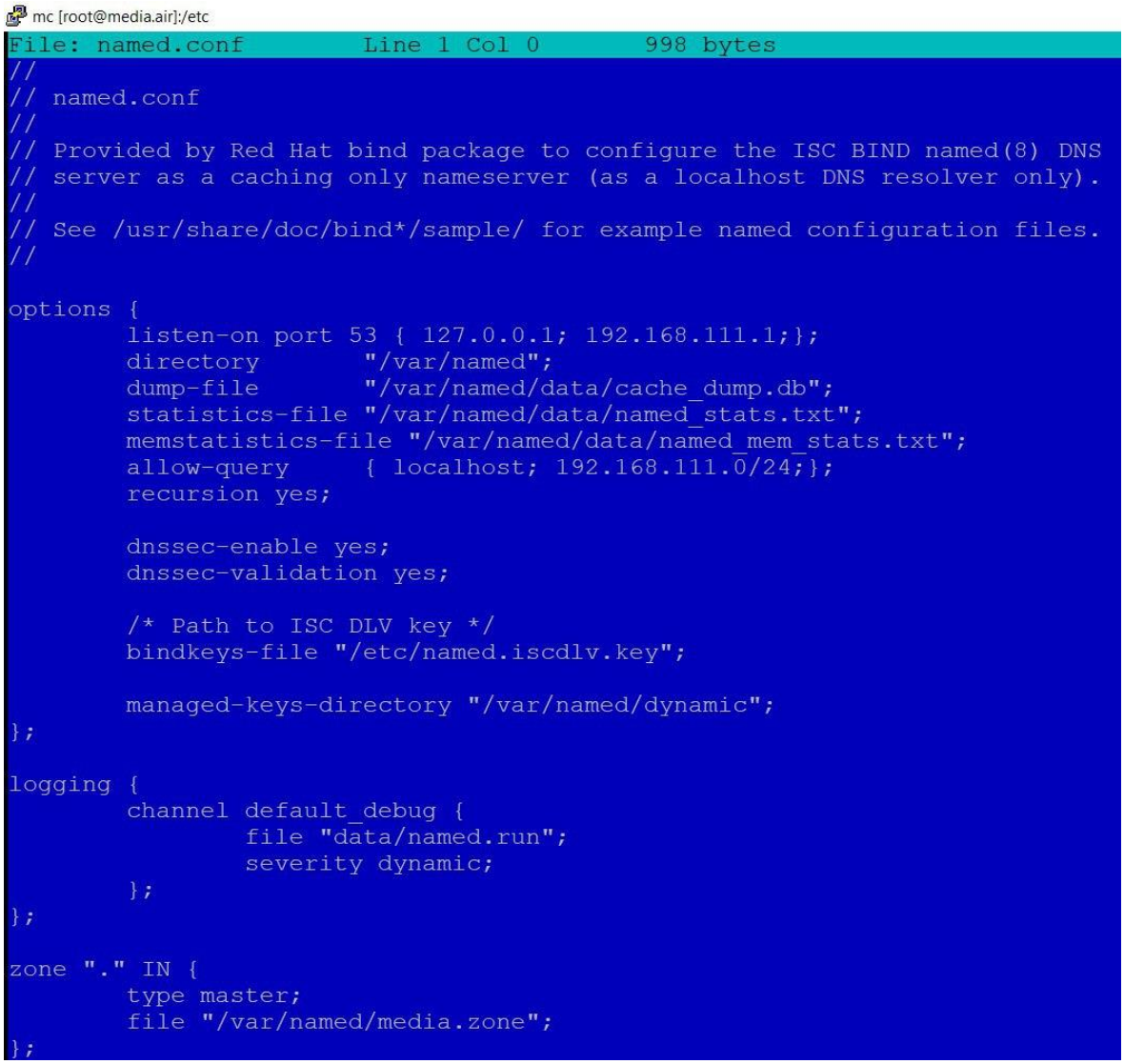


Figure 50. DNS tuning inside the configuration file of NAMED service

It was important to specify and configure the forward zone for server lookups. For that purpose, the dot-zone section was added to the named.conf file. Both created a master zone file for "example.com" address, but for the thesis project all requests by users had to navigate to the entertainment portal, that is why dot-zone was used. After all, there would

be no Internet access, and such a strategy would handle users' typos in the address bar. Both (2017) also specified a directory string in the section of options, but it was discovered that provision of the full path to the zone file in dot-zone section worked as well. The zone file, named media.zone was set in milliseconds and configured to translate all possible requests from users to address 192.168.111.1 – the server on which web application would run (Figure 51).

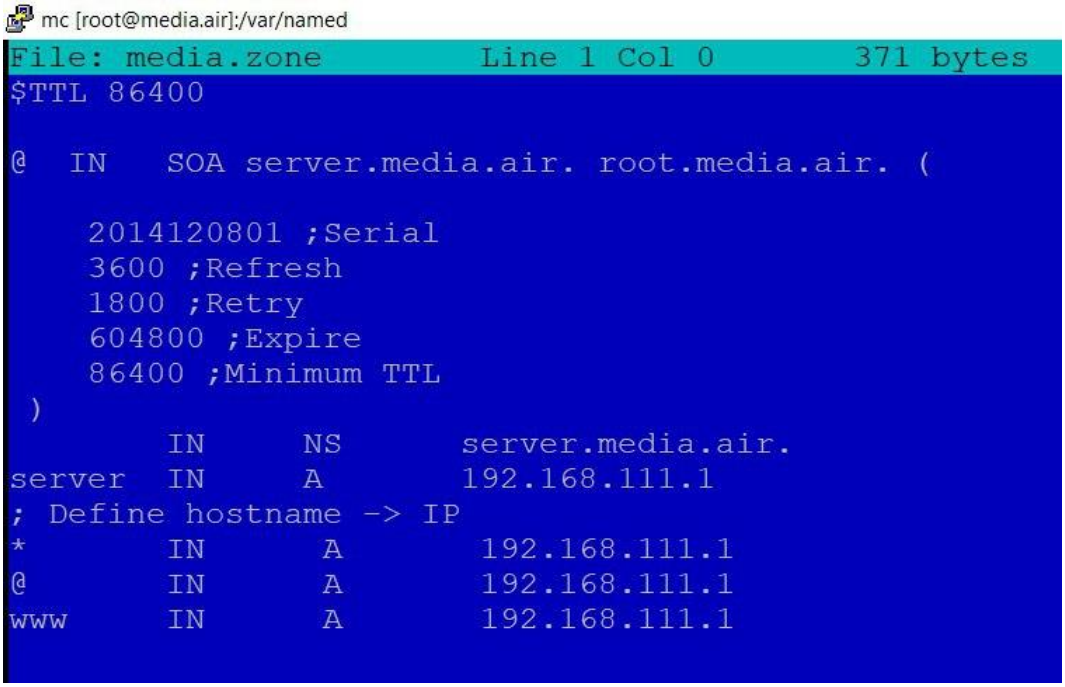


Figure 51. Forward lookup in media.zone

Having DHCP and DNS being setup, it was then important to configure the firewall of the server and open access to the services. Firewall file on CentOS6 software is called iptables and is located in /etc/sysconfig directory. The configurations set for the server could be observed in Figure 52.

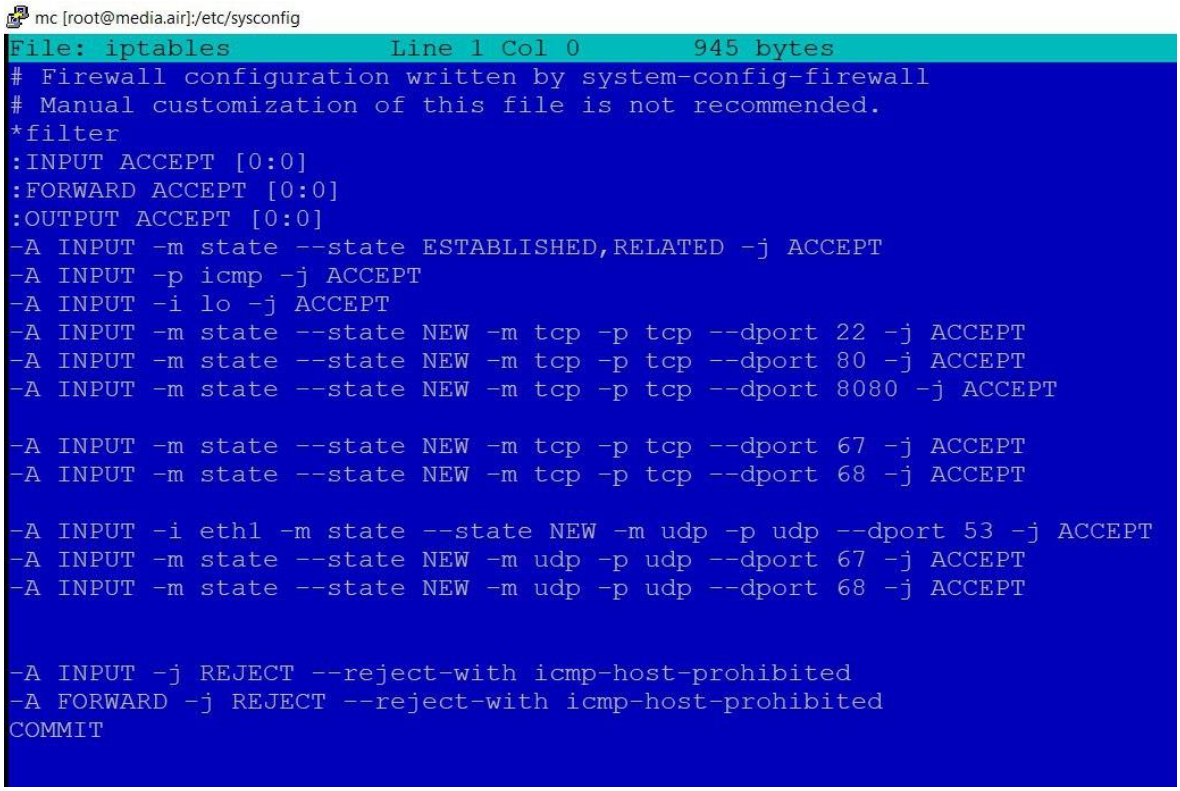


Figure 52. Iptables configuration with opening for needed ports

The first important ports opened, which was done prior to DHCP and DNS configurations, were ports 22 and 80 for TCP protocol. They were needed to establish an SSH, like for PuTTY, and Internet connection on the server. The next ports opened for requests of TCP origin were 8080, 67 and 68. Port 8080 was opened for the web application because Spring Boot projects run on that port by default. TCP ports 67 and 68 were needed DHCP server because these ports are used by devices for IP assignment requests and responses. Furthermore, the same ports for DHCP purposes were opened for UDP protocols, because it is another supported protocol for the service. Another UDP protocol requests were set for acceptance to port 53, which was dedicated to the DNS service. All other requests, including of forwarding nature, were set to rejection with ICMP host prohibition, which would result in service decline without explanation – a simple, yet very effective method against hijackers and people with intent to outflank the DNS service through direct provision of a resource IP address.

The final step in server configuration was setting the DHCP and DNS services for constant availability so that the services would be operating after every reboot of the system. Such a goal could be met with ntsysv utility installed before (Figure 53). By the simple call of the utility and tick at needed services, DHCP and DNS were set to automatic turn on after every restart of the wIFE server.

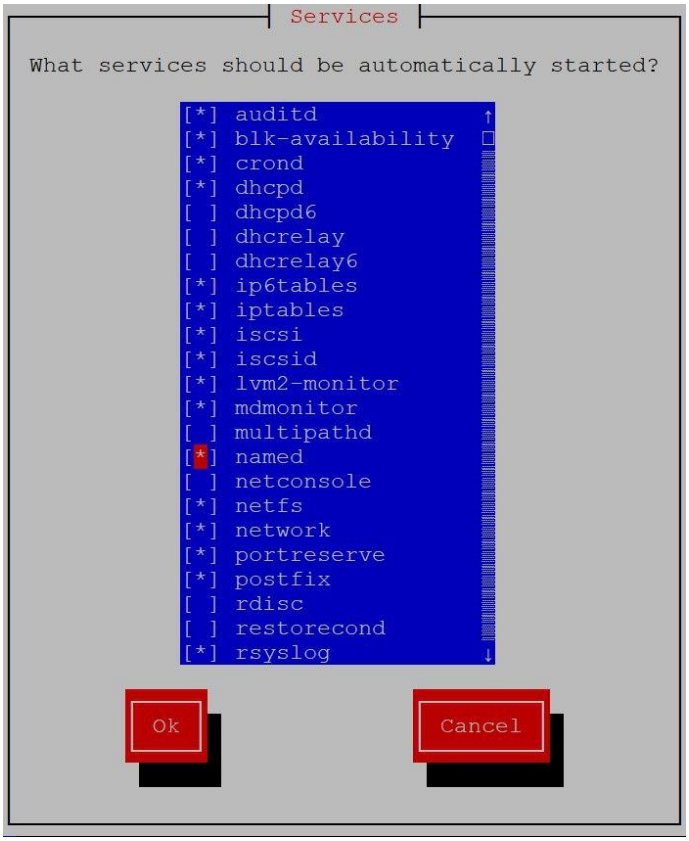


Figure 53. Enabling NAMED (DNS) and DHCPD (DHCP) services for automatic turn on

4.2.2 Configuration of wi-fi router

Final hardware element that required preparation was the wi-fi router. The router would create the network and would directly connect with the server. The router was also the only element missing at the beginning of the project and required separate acquisition measures. For the project, ASYS RT-N12 Wireless-N was purchased. The main reason for its selection was its cost affordability, and not its support for Repeater and Access Point functions. Most modern wi-fi routers, including the purchased one from ASUS, have a dedicated setting and configuration portal, accessible through a browser under LAN connection to a computer. The portal on the Asus router had different sections, most of which had to be checked for proper settings.

The first section observed was called Wireless. There, the name for the network was set to “media-air” and the network itself was turned into an open system (Figure 54).

Figure 54. Configuration of Wireless section in the portal of ASUS wi-fi router

In the LAN section, IP address of 192.168.2.1 and subnet mask of 255.255.255.0 were set (Figure 55). The address was set different to prevent possible conflicts with the server. The section had an important tab – DHCP server. DHCP on the router was enabled, so that connected users would obtain addresses from the router's network; DHCP server on CentOS6 was needed to provide addresses to wi-fi routers, that in theory had to be more than one. The range of IP address had to be in between 2 and 254. Lease time was set to standard 86400 seconds, which corresponded to 24 hours – 1 day. Another important thing to do was DNS Server specification, which was set to 192.168.2.1 – the router itself; the actual DNS would still come from the CentOS6 server (Figure 56).

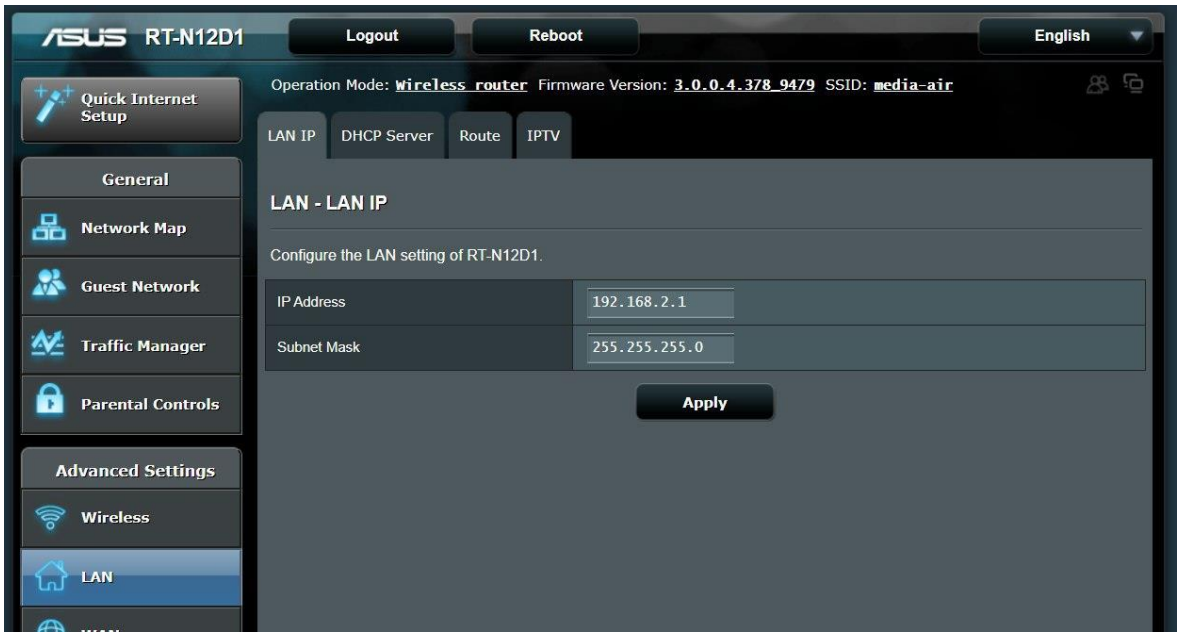


Figure 55. Setting address and subnet mask for the router

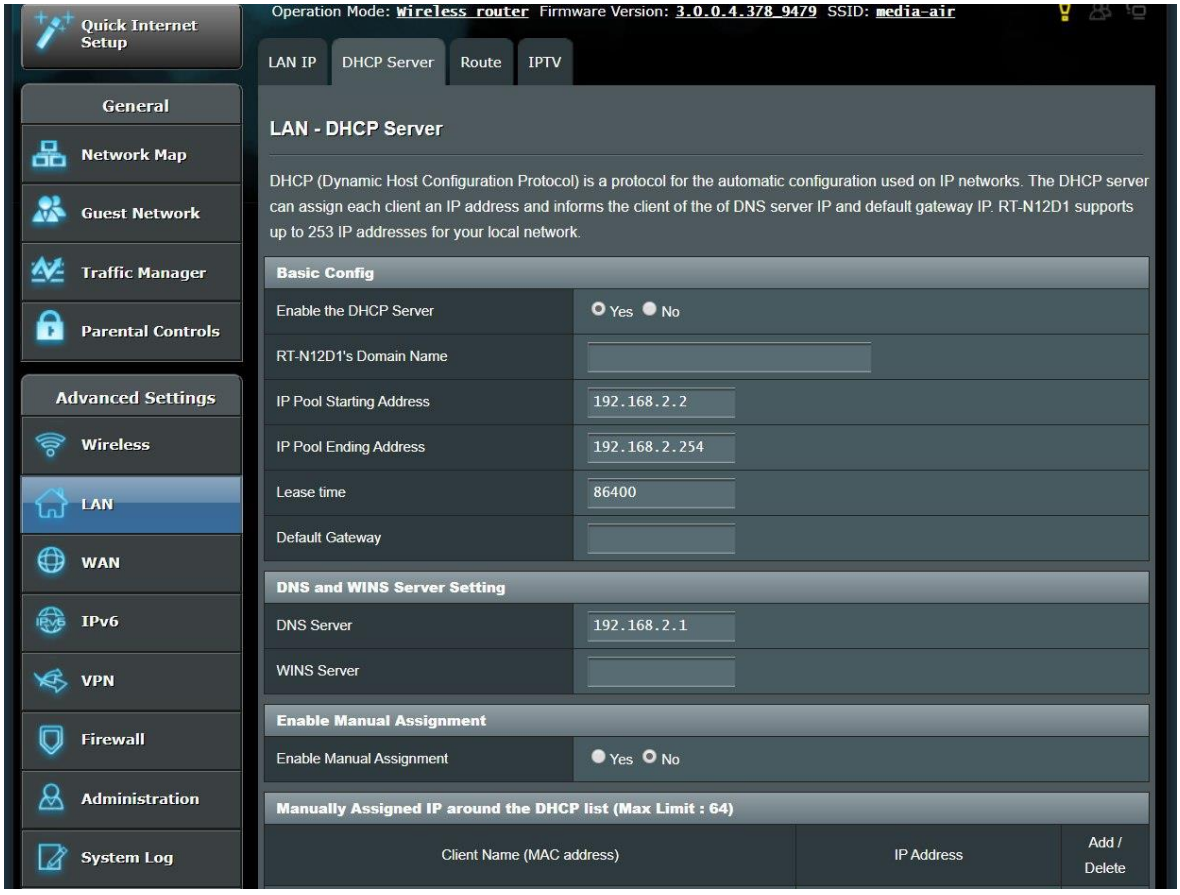


Figure 56. Let wi-fi router assign IP addresses to connected users. Specification of DNS on the router itself

The DNS service on the router was directed to the router itself because the router would obtain DNS service from the virtualized CentOS6 server. To make it work, the router had

to obtain its IP address on the WAN port automatically (Figure 57). This condition was set in the WAN section of the configuration portal. NAT – network address translation, was also enabled. The network was also set to require no authentication for connected users.

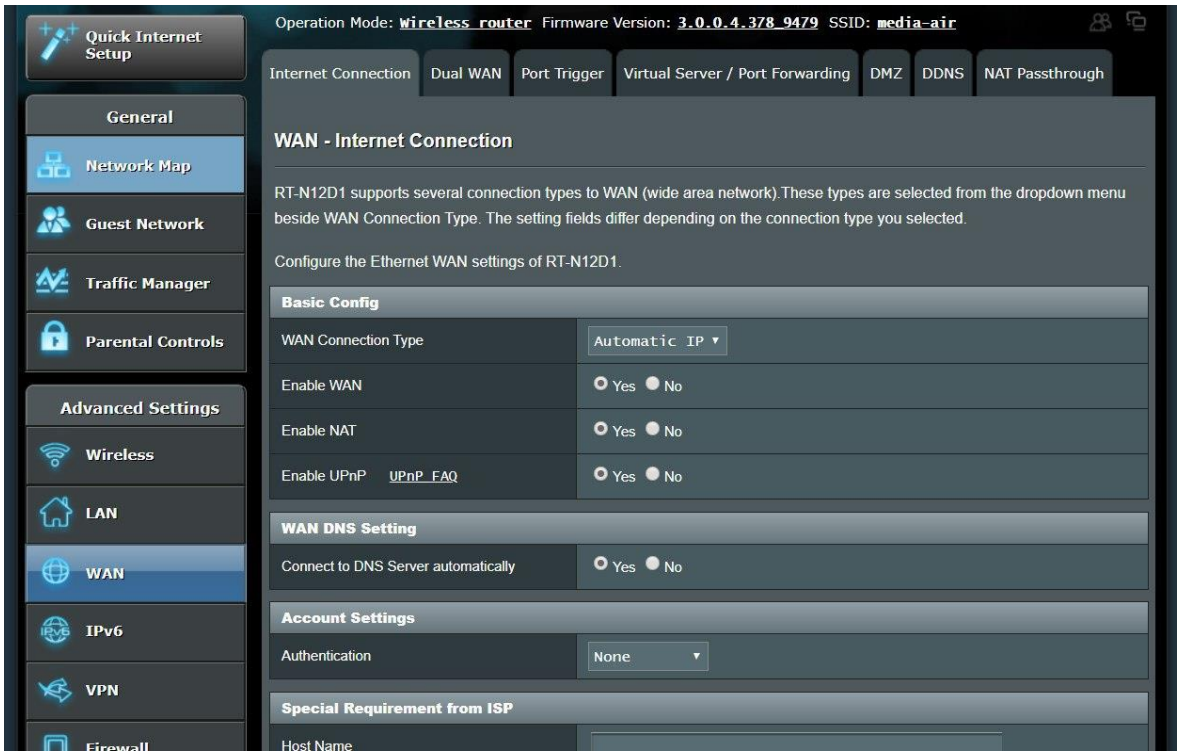


Figure 57. The IP address on WAN port was set to automatic to provide besides all DNS service from the server

Applied actions in other sections aimed at disabling IPv6 and VPN. Firewall settings of the router were not changed. Time zone was set to Helsinki time. The router was kept operating in wireless router mode.

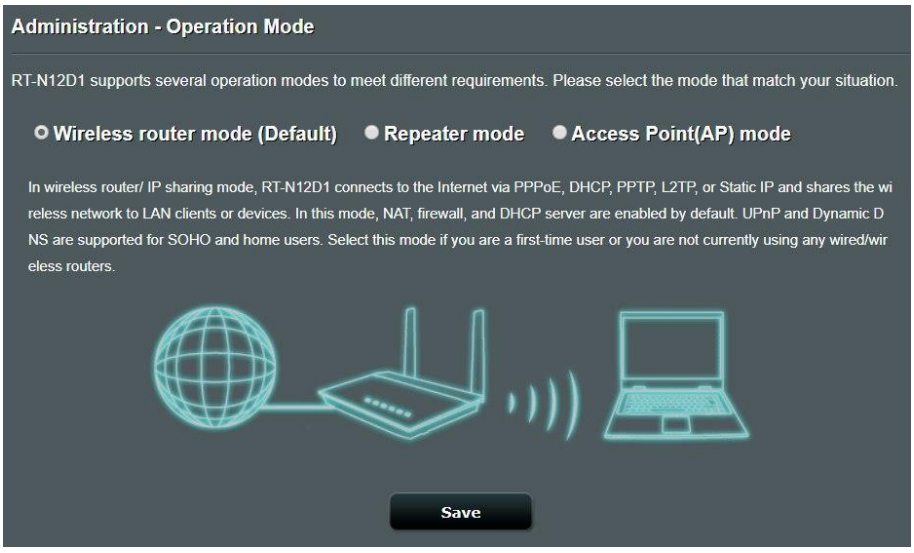


Figure 58. Keeping of wi-fi router to operate in default mode

In the final form, the wi-fi router had to be connected with the laptop, on which the server was virtualized, by WAN port. The server – CentOS6 instance with DHCP and DNS, through bridge connection was using the Ethernet interface of the host system – Windows 10 laptop. Wi-fi router was establishing a WAN connection to the Ethernet interface of the laptop, where it would be exposed to the CentOS6 instance. Since WAN connection type of the router was set automatic, DHCP service of the server would assign an IP address and provide DNS to the wi-fi router. Thus, when a user would connect to the wi-fi network, called “media-air”, obtained IP address would originate from the router. When a user would search for a resource in a web browser, wi-fi router would use DNS from the server to determine the actual address of a resource. The server would point all requests to the server itself, where the web application would be hosted. Thus, users would access the entertainment portal of the designed wireless IFE under any address query, as long as wi-fi connection is kept.

4.2.3 Deployment of web application on the server

The final step in the creation of the wireless IFE was the deployment of the web application to the server. For that purpose, a media-air folder was created in the root directory of the CentOS6 instance. The folder also included a Media directory, where multimedia materials, destined for Movies, Songs and Series, had to be stored. Movies folder had separation of poster images and trailer videos.

Since the backend of the application was initially developed and tested on a windows device, changes had to be applied to provide proper file read logic in a Linux environment. Besides, it was found that the default use of port 8080 resulted in constant port typing to access the portal of the deployed application on the server. When the port on the backend was changed to 80 instead, the issue was gone. The final modification was made on the frontend part of the application. MoviesHttpService had to use the static address of the deployment server – 192.168.111.1 (Figure 17). After all these changes applied, the web application was rebuilt and deployed to the server.

4.3 Testing of the wIFE system

Upon completion of wireless IFE development, that included the creation of software and tuning of hardware, the performance of user tests was possible. Tests were done on laptops and smartphones. The system worked on personal laptops and mobile devices of Android operating systems: it was possible to access the portal and watch movie trailers. Mobile devices from Apple Inc. did not allow to stream video content from browser: the

portal was opening, however, upon pressing a play button in a video frame, nothing happened; logs from the server indicated requests on video provision endpoint. Attempts to investigate the problem was not possible, because the iPhone device was not in the possession of the developer and could not be taken for sufficient amount of time to discover the rooting cause of the issue.

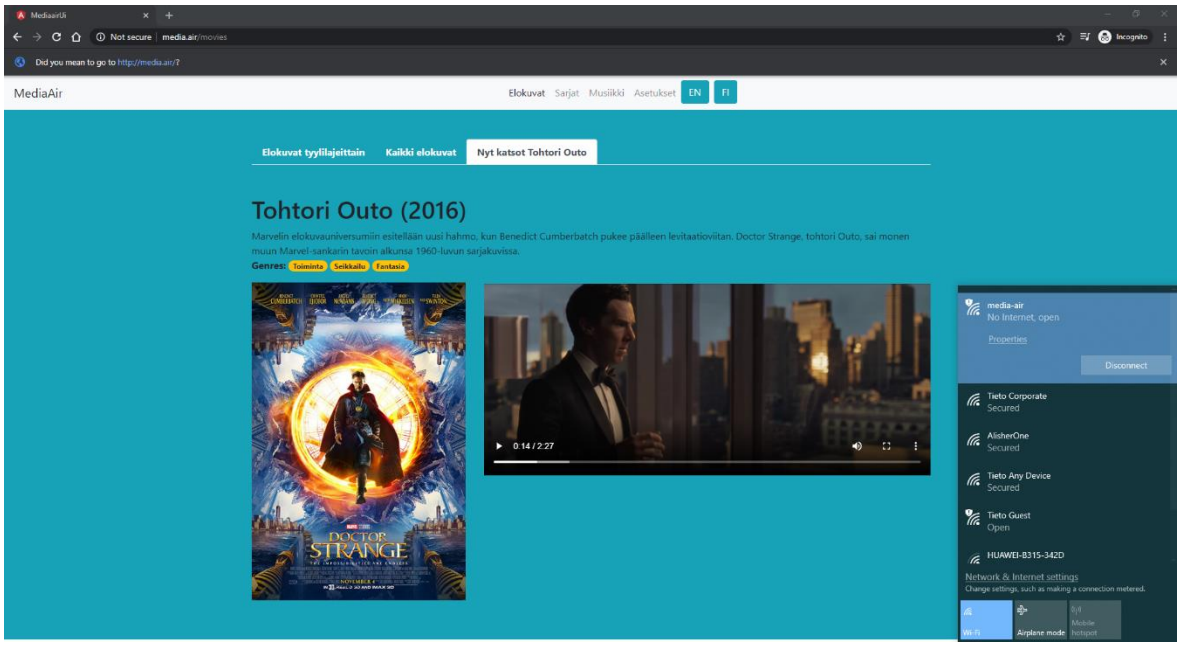


Figure 59. wIFE portal interface under connection to the network

One particular problem with the access of the portal was observed in the use of the desktop version of Chrome web browser from Google. Apparently, the browser used internal DNS or a caching mechanism that tried to resolve the domain on the Internet, which resulted in service deny due to Internet connection absence (Figure 60). In order to access the portal on the Chrome browser, it was necessary to provide the full address of the IFE: http://media.air.

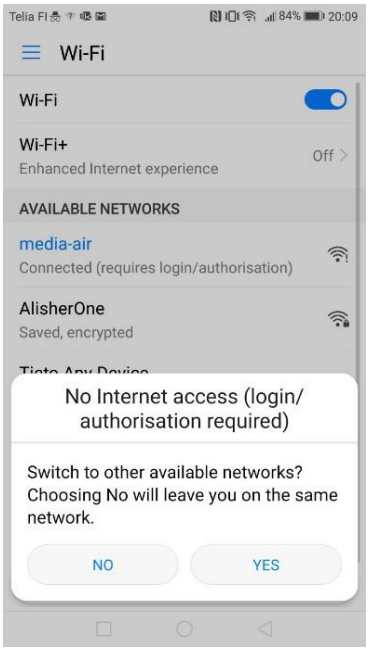


Figure 60. Connection to the system did not provide Internet access

In overall, however, the system worked as planned. Portal was accessible on all devices. The interface was simple and fluid to resolutions of devices' screens. Devices, upon connection to the wi-fi network, were obtaining IP addresses of the router's network (Figure 61). The application was able to read the requested multimedia files and show them to users. Instant translation of the interface between Finnish and English languages worked even better than planned: the translation was permissible even without the need for stopping streaming of videos.

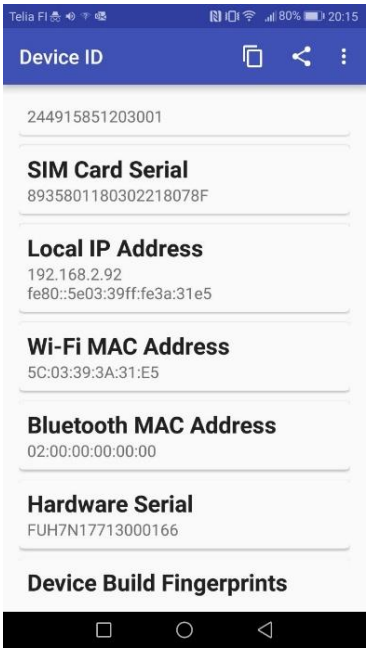


Figure 61. Obtained IP address belonged to wi-fi router's network

5 Evaluation and discussion

Evaluation of the performed work and thesis itself through a reflection of the project is another crucial step. It allows to highlight the progress of the work and point to aspects that either accelerated or negatively impacted the development of the project. Though the thesis work started in summer 2019, it was finalized by February 2020. An important clarification should be added, that even though theoretical part of the project was provided in the first part of the report, actual analysis of the technology as a key attribute of cabin décor for airlines to attract customers was performed after the software of the project was finished and in parallel to the system configuration. Evaluation of scientific and published works regarding the importance of IFE systems stood well in contrast to initial personal opinion. Yet it still crucial to convey unbiased analysis over the technology's significance.

The major reflection could be performed over the empirical part of the project, i.e. development of the software and configuration of the hardware into an operating wireless entertainment system. It can be concluded that IT projects could hardly be done rapidly alone. The main cause for the software and system development spanning for six months was the involvement of one developer only. In case the project was developed under the involvement of a team of developers, the final product could have been obtained by the middle of Autumn 2019. Nonetheless, the inclusion of the initial plan and scoping of the IT project with proper boundaries allowed to successfully finish it. Furthermore, use of Git technology, for version control in web application development part provided a solid tool to track the progress and apply many new features, unknown to the developer before, onto the project, without threatening the state of stable parts. For instance, use of BehaviorSubject and RxJS package in whole was not done prior to the project. The technology was studied and applied for the needs of REST API communication between backend and frontend, instant localization of the interface. Numerous approaches were applied for such functionality of the UI, many of which failed. Without the use of Git, the project would have stuck or delayed for a much longer period.

The notion of planning and scoping of IT projects is also worthy of mentioning. Even though, arguably the developed system was very limited in terms of features and capabilities: no internet connection, no security measures, access only to video trailers as entertainment source – without setting of limitations on the planned project, development of the system would require additional time. Many areas of services for proper IFE system require the application of technological know-how simply unknown to the author. Only through thorough studies of those technological methods, the significantly advanced system could have been produced. However, the project consumed enough amount of

time in researching on video streaming in web applications, integration of Angular with Spring Boot and set up of a wi-fi network with access to services run on a connected to it server.

Outcomes in terms of knowledge obtained is a positive aspect of the project. Again, particular areas in system and software engineering had to be researched, understood and applied. Often, materials and tutorials found on the Internet could not provide a direct solution to the aims of the project. Nonetheless, understanding of the material and technology gave a path to directions of set functionality achievement: integration of the frontend and backend part was developed with the personally discovered method, which would not be derived unless tutorials and documentation of Spring Boot and Angular were studied. A similar situation was experienced in the system and hardware configuration part of the thesis work, where DHCP and DNS configurations were done in a slightly different way than in analyzed tutorial materials.

Separate reflection could be made on theoretical framework studied for the project. As it was mentioned before, the analysis showed a significant miscorrelation of expert consensus and initial personal opinion towards the technology. The project itself was formed after a series of personal negative experience with European airlines. However, the research consensus derived after studies of a considerable number of credible sources revealed the relative insignificance of IFE systems, especially on flights short- and medium-time duration. Nonetheless, it should be added that few papers available directly studied the importance of IFE for passengers. Once again it is necessary to point out that only Alamdari was concern in the technology's importance. Her work, however, was done in 1999, when IFE systems were still robust and technologically limited.

Presumably, the thesis work could have involved populace survey regarding IFE systems and their value creation for passengers. Such work would have required additional effort and time, but it could significantly decrease the degree of uncertainty around the topic. An important aspect of such work would be a classification of travellers by their socio-demographic statuses and application of credible statistical approaches in the evaluation of data at the analysis stage.

6 Thesis conclusion

In conclusion, it can be said that the planned system, that would resemble a functioning wireless in-flight entertainment system, was successfully achieved. The system was developed under limited budget with devices available at possession on commencing stage of the project (except for the wi-fi router, that was purchased). Developed software and configured hardware worked in pair to create a wi-fi network, upon connection to which user could navigate to the entertainment portal and enjoy multimedia content available. The planning stage of the project involved a thorough scoping of the project, that left only a few services and capabilities the system had to provide. Though initially it was considered to be excessively short, at the implementation stage it was revealed that attainment of even the limited state of functionality of the system was a challenging task in terms of time needed for the study of credible materials and application of the methods in the development of the system. Provision of additional services, like music entertainment, cabin class division and interface for IFE management would have postponed the overall progress of the thesis.

Nonetheless, the time constraint and other limitations still affected the overall result. As it was mentioned the IFE could not play the video material on iOS devices, that could not be investigated further due to the lack of devices for development and project time constraint. It was also revealed that the internal caching of Chrome browser resulted in full address specification for portal access. While other browsers did not result in a similar outcome, additional work could have been done to tackle the issue. Moreover, the system developed was different from actual IFE products in a physical way. It was stated in the empirical part of the report that a cabin-compatible solution would require additional work and investment into hardware design of the interior of aircrafts used for commercial transportation of humans. Knowledge access and budget constraints were the main factors that prevented such endeavour.

Several conclusions can be made about in-flight entertainment systems and its industry in general. The aviation market has been soaring in terms of global profits. However, fertile medium in the air linked to the legislative liberalization measures, aircraft technological advancements and overall global economic improvement led to the proliferation of the industry with a diverse number of airlines and as a result fierce competition. In the face of the tight competition, airlines could use in-flight entertainment to provide additional value and comfort for passengers and thus win over travellers' choice. Although analysis of the published sources revealed the low significance of the technology for passengers, its provision could play an important role in long-haul flights. Besides, no recent research

material was obtained during the project work, that would target the IFE importance in particular. It can be assumed, that opinion of travellers has changed over time and new research need to be conducted to evaluate the opinion of airline passengers regarding the presence or absence of in-flight entertainment onboard.

Wireless in-flight entertainment could be the most suitable solution for airlines that are concerned regarding service cost. Available commercial solutions revealed to be many times affordable than standard IFE systems. Moreover, though the thesis work did not involve the creation of a system that could be installed on an airplane, it was showed that it is possible to create a proprietary solution. Assuming the wider budget allocation capabilities of airlines worldwide, air transit companies could develop and install wireless IFE systems on their own. Such systems would not overload the plane and at the same time give access to numerous entertainment materials for passengers through their devices.

The final conclusion regarding the overall thesis work can be projected on the significant professional improvement the project gave. Due to the nature of the system development, skills in software and system engineering, a better understanding of hardware use and acquaintance with new technologies were obtained. Regardless of the final product’s limited capabilities, thesis project resulted in a high contribution to expertise increase in the information technology sphere.

7 Recommendations for further works

Further work could be done attempts to create an actual wireless in-flight entertainment system. Hardware, software and system aspects could be improved. The recommended system could be based on the work done in this project but include a hardware design suitable for airplane interior. In that case, a future project should involve cooperation with local airlines and manufacturers. Software-wise further improvements could be added to the functionality of the developed software: additional of other entertainment sources, like music, flight status and video games; provision of in-flight shopping and onboard service request through the application; authorization of users to determine their fare class and thus provision of fare-unique services. System-wise additional routers could be added. The server could enforce security measures to prevent malicious attacks on the system from connected users. Internet provision could also be added since its accessibility is highly valued by travellers. Moreover, the system could include a design for fast upgrade and content management, especially at a short period of time when planes are on the ground.

Improvements in the testing method of the system could also be applied. Through cooperation with airlines, it could be possible to test such product onboard a plane with actual airline passengers. Both technical and survey results from passengers could be obtained for work and system analysis. Mentioning surveying, it could be added that with the additional budget proper research could be done in the determination of IFE importance for passengers. Such work could be done as a separate thesis project, abstracted from the development aspect of the IFE product. It would require a properly planned budget for surveying of travellers and application of different mathematical and statistical models for analysis, evaluation and scenario simulations.

References

Airbus SE 2018. A330-900 in formation flight with A350-1000. URL: https://www.airbus.com/search.image.html?tags=products-and-solutions%3Acommercial-aircraft%2Fa350-family%2Fa350-1000&page=1&results_page_index=2&lang=en&tagLogicChoice=OR#searchresult-image-all-70. Accessed: 5 March 2020.

Alamdari, F. 1999. Airline in-flight entertainment: the passengers' perspective. Journal of Air Transport Management, 5, 4, pp. 203-209.

Asab, M. 2019. Integrating an Angular project with Spring Boot. URL: https://medium.com/@majdasab/integrating-an-angular-project-with-spring-boot-e3a043b7307b. Accessed: 5 March 2020.

Bachman, J. 2014. Airlines Need You (and Your Gadgets) to Kill Those Seat-Back Screens. Bloomberg Businessweek. URL: https://www.bloomberg.com/news/articles/2014-09-03/byod-airlines-need-your-carry-on-gadgets-to-end-seat-back-screens. Accessed: 5 March 2020.

Both, D. 2017. Build your own DNS server on Linux. URL: https://opensource.com/article/17/4/build-your-own-name-server. Accessed: 5 March 2020.

Cederholm, T. 2014. Low-entry barriers intensify competition in airline industry. URL: https://marketrealist.com/2014/12/low-entry-barriers-intensify-competition-airline-industry. Accessed: 5 March 2020.

Dow Jones & Company Inc 2010. PRESS RELEASE: Alaska Airlines Now Offering Inflight Wi-Fi on More Than Half Its Aircraft. Dow Jones Institutional News.

Eljaiek, E. 2017. Spring Boot with AngularJS html5Mode. URL: https://stackoverflow.com/questions/24837715/spring-boot-with-angularjs-html5mode/44850886#44850886. Accessed: 5 March 2020.

European Commission 2017. International aviation: United States. URL: https://ec.europa.eu/transport/modes/international-aviation-united-states_en. Accessed: 5 March 2020.

Federal Aviation Administration 2017. A Brief History of the FAA. URL: https://www.faa.gov/about/history/brief_history. Accessed: 5 March 2020.

GitHub, Inc. 2013. frontend-maven-plugin. URL: https://github.com/eirslett. Accessed: 5 March 2020.

Goel, N. 2019. Are Angular applications SPAs?. URL: https://codeburst.io/angular-spas-168a94a0959a. Accessed: 5 March 2020.

GOGO LLC 2016. Gogo Surpasses 2,000 Gogo Vision Equipped Aircraft. URL: http://concourse.gogoair.com/gogo-surpasses-2000-gogo-vision-equipped-aircraft. Accessed: 5 March 2020.

GOGO LLC 2016. In-cabin network. URL: https://www.gogoair.com/commercial/inflight-systems/in-cabin-network. Accessed 5 March 2020.

GOGO LLC 2017. Designing for the next generation of IFE. URL: http://concourse.gogoair.com/gogo-surpasses-2000-gogo-vision-equipped-aircraft. Accessed 5 March 2020.

GOGO LLC 2018. 2018 Global Traveler Research Study. URL: https://www.gogoair.com/learning-center/2018-global-traveler-research-study. Accessed 5 March 2020.

Google 2015. Component. URL: https://angular.io/api/core/Component. Accessed: 5 March 2020.

Google 2015. Introduction to Angular concepts. URL: https://angular.io/guide/architecture. Accessed: 5 March 2020.

Google 2015. Introduction to components and templates. URL: https://angular.io/guide/architecture-components. Accessed: 5 March 2020.

Gourdin, K. N. 2015. A Profile of the Global Airline Industry. Business Expert Press. New York.

Hunt, J. & Truong, D. 2019. Low-fare flights across the Atlantic: Impact of low-cost, long-haul trans-Atlantic flights on passenger choice of Carrier. Journal of Air Transport Management, 75, pp. 170-184.

IATA, 2015. Annual Review 2015. URL: https://www.iata.org/contentassets/c81222d96c9a4e0bb4ff6ced0126f0bb/iata-annual-review-2015.pdf. Accessed: 5 March 2020.

IATA, 2016. Annual Review 2016. URL: https://www.iata.org/contentassets/c81222d96c9a4e0bb4ff6ced0126f0bb/iata-annual-review-2016.pdf. Accessed: 5 March 2020.

IATA, 2017. Annual Review 2017. URL: https://www.iata.org/contentassets/c81222d96c9a4e0bb4ff6ced0126f0bb/iata-annual-review-2017.pdf. Accessed: 5 March 2020.

IATA, 2018. Annual Review 2018. URL: https://www.iata.org/contentassets/c81222d96c9a4e0bb4ff6ced0126f0bb/iata-annual-review-2018.pdf. Accessed: 5 March 2020.

IATA, 2019. Annual Review 2019. URL: https://www.iata.org/contentassets/c81222d96c9a4e0bb4ff6ced0126f0bb/iata-annual-review-2019.pdf. Accessed: 5 March 2020.

IATA, 2019. IATA forecasts $29.3bn net profit for airlines in 2020. URL: https://www.airlines.iata.org/news/iata-forecasts-293bn-net-profit-for-airlines-in-2020. Accessed: 5 March 2020.

Java Developer Zone 2017. Spring boot index page example. URL: https://javadeveloperzone.com/spring-boot/spring-boot-index-page-example/. Accessed: 5 March 2020.

Kallonen, I. 2017. Gathering user insights to drive the design of an airplane cabin for Northeast Asia. Master’s Thesis. Aalto University, School of Engineering. URL: https://aaltodoc.aalto.fi/handle/123456789/24443. Accessed: 5 March 2020.

Kili, A. 2018. How to Setup DHCP Server and Client on CentOS and Ubuntu. URL: https://www.tecmint.com/install-dhcp-server-client-on-centos-ubuntu/. Accessed: 5 March 2020.

Korhonen, P. 2019. Developing the Inflight Customer Experience of Airline X. Master's Thesis. Haaga-Helia University of Applied Sciences, Degree Programme in Aviation and Tourism Business. URL: https://www.theseus.fi/handle/10024/263519. Accessed: 5 March 2020.

Ku, C. 2019. APEX's ARC Committee Spreads the Word About In-Flight Advertising. URL: https://apex.aero/2019/11/20/arc-advertising-glossary. Accessed: 5 March 2020.

McCabe, R. M. 2006. Airline Industry Key Success Factors. URL: https://gbr.pepperdine.edu/2010/08/airline-industry-key-success-factors. Accessed: 5 March 2020.

Medina-Muñoz, D. R., Dolores, M.-M. R. & Ángel, S.-C. M. 2018. Determining important attributes for assessing the attractiveness of airlines. Journal of Air Transport Management, 70, pp. 45-56.

Milioti, C. P., Karlaftis, M. G. & Akkogiounoglou, E. 2015. Travel perceptions and airline choice: A multivariate probit approach. Journal of Air Transport Management, 49, pp. 46-52.

MojoHaus 2017. Usage. URL: https://www.mojohaus.org/exec-maven-plugin/usage.html. Accessed: 5 March 2020.

Mozilla 2019. MVC. URL: https://developer.mozilla.org/en-US/docs/Glossary/MVC. Accessed: 5 March 2020.

OECD 2014. Airline Competition. URL: http://www.oecd.org/officialdocuments/publicdisplaydocumentpdf/?cote=DAF/COMP(2014)14&docLanguage=En. Accessed: 5 March 2020.

Oracle 2007. Welcome to VirtualBox.org!. URL: https://www.virtualbox.org/. Accessed: 5 March 2020.

Oracle 2011. How to download and install prebuilt OpenJDK packages. URL: https://openjdk.java.net/install/. Accessed: 5 March 2020.

Oxford Business Group 2018. Rapid expansion of global air travel industry propels investment. The Report: Mexico 2018. p. 316. URL: https://oxfordbusinessgroup.com/overview/skybound-rapid-expansion-global-industry-propels-investment-efforts. Accessed: 5 March 2020.

Pallini, T. 2020. Airbus recently delivered the 350th A350 plane, its answer to Boeing's revolutionary 787 Dreamliner. Here's how the new aircraft is reshaping air travel. URL: https://www.businessinsider.com/history-of-airbus-a350-xwb-next-generation-widebody-plane-2020-2?r=US&IR=T. Accessed: 5 March 2020.

Panasonic Avionics Corporation 2019. NEXT Series. URL: https://www.panasonic.aero/our-offerings/systems/next-series. Accessed: 5 March 2020.

Pierbon, M. 2019. Personal electronic devices: Latest trend in inflight entertainment. URL: https://www.aviationbusinessnews.com/cabin/ife-connectivity/inflight-entertainment-personal-electronic-devices. Accessed: 5 March 2020.

Pivotal 2020. Common Application properties. URL: https://docs.spring.io/spring-boot/docs/current/reference/html/appendix-application-properties.html#templating-properties. Accessed: 5 March 2020.

Prasad, S. 2018. Building a Web App Using Spring Boot, Angular 6, and Maven. URL: https://dzone.com/articles/building-a-web-app-using-spring-boot-angular-6-and. Accessed: 5 March 2020.

PuTTY project 2018. Download PuTTY. URL: https://www.putty.org/. Accessed: 5 March 2020.

Qatar Airways 2020. In-Flight Entertainment. URL: https://www.qatarairways.com/en-fi/ife-homepage.html. Accessed: 5 March 2020.

Ramsey, J. 2011. In-Flight Entertainment. Avionics Magazine, 35, 9.

Red Hat, Inc. 2006. ntsysv. URL: http://web.mit.edu/rhel-doc/5/RHEL-5-manual/Deployment_Guide-en-US/s1-services-ntsysv.html. Accessed: 5 March 2020.

Schawalder, J. 2014. The Future of Inflight Entertainment in Europe, According to Passenger Expectations: Why Airlines Should Embrace Consumer Technology. Anchor Academic Publishing. Hamburg.

Shashank, N. 2013. Social Evolution. Airline Business, 29, 3, p. 41.

Singapore Airlines 2020. Inflight Entertainment. URL: https://www.singaporeair.com/en_UK/kr/flying-withus/entertainment. Accessed: 5 March 2020.

Syer, D. 2014. Java Spring Boot: How to map my app root (“/”) to index.html?. URL: https://stackoverflow.com/questions/27381781/java-spring-boot-how-to-map-my-app-root-to-index-html. Accessed: 5 March 2020.

Tarry, C. 2019. First-class service or false economy?. Airline Business, 35, 3, p. 50.

Taylor, S. 2019. 40 Success Stories: When Kontron Streamlined IFEC Hardware. URL: https://apex.aero/2019/08/03/40-success-stories-kontron-ifec-hardware. Accessed: 5 March 2020.

Taylor, S. 2019. 40 Success Stories: When Lufthansa Systems Premiered Wireless IFE. URL: https://apex.aero/2019/08/15/40-success-stories-lufthansa-systems-boardconnect-wireless-ife. Accessed: 5 March 2020.

Taylor, S. 2019. What’s the Deal with Wireless IFE? Bluebox Aviation Explains. URL: https://apex.aero/2019/08/03/apex-insider-video-wireless-ife-bluebox. Accessed: 5 March 2020.

The Apache Software Foundation 2020. What is Maven. URL: https://maven.apache.org/what-is-maven.html. Accessed: 5 March 2020.

The Boeing Company 2019. First of Many: 787 Dreamliner Celebrates 10 Years Since First Flight. URL: https://www.boeing.com/features/2019/12/787-1st-flight-anniversary-12-19.page. Accessed: 5 March 2020.

The CentOS Project 2019. The CentOS Project. URL: https://www.centos.org/. Accessed: 5 March 2020.

The Emirates Group 2018. Inflight entertainment. URL: https://www.emirates.com/fi/english/experience/inflight-entertainment. Accessed: 5 March 2020.

Tolpa, E. 2012. Measuring customer expectations of service quality: case airline industry. Master's Thesis. Aalto University, Department of Information and Service Economy. URL: https://aaltodoc.aalto.fi/handle/123456789/3904. Accessed: 5 March 2020.

Tozer-Pennington, V. 2019. The Aviation Industry Leaders Report 2019: Tackling headwinds. Aviation News Ltd. Stoke-On-Trent. URL: https://assets.kpmg/content/dam/kpmg/ie/pdf/2019/01/ie-aviation-industry-leaders-report-2019.pdf. Accessed: 5 March 2020.

Trac 2020. Welcome to the Midnight Commander Development Center. URL: https://midnight-commander.org/. Accessed: 5 March 2020.

Tsai, W.-H. et al. 2014. A green approach to the weight reduction of aircraft cabins. Journal of Air Transport Management, 40, pp. 65-77.

w3resource 2020. Getting started with Angular. URL: https://www.w3resource.com/angular/getting-started-with-angular.php. Accessed: 5 March 2020.

Vaishnavi, M. R. 2020. Top 10 Java Frameworks You Should Know. URL: https://www.edureka.co/blog/java-frameworks/. Accessed: 5 March 2020.

VMWare Inc. 2015. Spring Boot. URL: https://spring.io/projects/spring-boot. Accessed: 5 March 2020.

WNS Global Services 2017. Top trends shaping the airline industry. URL: https://www.wns.com/insights/articles/articledetail/459/top-trends-shaping-the-airline-industry. Accessed: 5 March 2020.